Răzvan Rughiniș
Răzvan Deaconescu
Andrei Ciorba
Bogdan Doinea

Computer Science

Rețele locale

Editura PRINTECH

Material cu drept de autor

0 Introducere în sisteme de operare ... 1
0.1 Ce este un sistem de operare? ... 1
0.1.1 Ce înseamnă administrarea unui sistem de operare? ... 2
0.2 Noțiuni de bază pentru administrarea unui sistem de operare Linux ... 2
0.2.1 Componentele unui sistem GNU/Linux. Distribuții ... 3
0.2.2 Sistemul de fișiere ... 3
0.2.3 Gestiunea utilizatorilor ... 4
0.2.4 Drepturi pe sistemul de fișiere ... 5
0.2.5 Gestiunea pachetelor ... 6
0.2.6 Gestiunea pachetelor DEB ... 7
0.2.7 Gestiunea serviciilor ... 8
0.2.8 *Shell scripting* ... 9
0.3 Introducere în Microsoft Windows Server 2008 ... 12
0.3.1 Despre Microsoft *Windows Server 2008* ... 12
0.3.2 *Windows PowerShell* ... 13
1 Nivelul fizic ... 32
1.1 Semnale ... 32
1.1.1 Tipuri de semnale ... 32
1.1.2 Codarea ... 34
1.1.3 Modularea ... 36
1.1.4 Multiplexarea ... 37
1.1.5 Caracteristici ale semnalului ... 38
1.2 Soluții de comunicație pe cupru ... 41
1.2.1 Cablul coaxial ... 41
1.2.2 Cablul torsadat ... 41
1.3 Soluții de comunicație pe fibră optică ... 46
1.3.1 multi-mode ... 48
1.3.2 single-mode ... 48
1.3.3 Comparație între single-mode și multi-mode ... 48
1.3.4 Mod de construcție, conectori ... 49
1.3.5 Multiplexarea prin divizarea lungimii de undă – WDM ... 52
1.3.6 Comparație între fibra optică și cablul UTP ... 53
1.4 Caracteristici ale mediilor de transmisie ... 54
1.4.1 Frecvența ... 54
1.4.2 Lățimea de bandă ... 54
1.4.3 Unități de măsură ... 54
1.4.4 Baseband și broadband ... 55
1.5 Echipamente de rețea de nivel fizic ... 56
1.5.1 Repetorul ... 56
1.5.2 Convertorul ... 58
1.6 Studii de caz ... 58
1.6.1 Realizarea *patch*-urilor UTP *straight-through, crossover, rollover* ... 58
2 Rețele Ethernet ... 60
2.1 Noțiuni generale ... 60
2.1.1 Structura cadrului Ethernet ... 61
2.1.2 CSMA/CD ... 62
2.1.3 Full-duplex Ethernet ... 64
2.2 Ethernet switching ... 65
2.2.1 Învățarea adreselor ... 66
2.2.2 Deciziile de comutare ... 67
2.2.3 Evitarea buclelor de nivel doi – STP ... 68
2.2.4 Metode de comutare ... 73
2.2.5 Switch vs. Bridge ... 74

Material cu drept de autor

2.3 Rețele locale virtuale ... 75
2.3.1 Tipuri de VLAN-uri ... 78
2.3.2 Legături acces/trunchi ... 79
2.3.3 Metode de identificare ... 79
2.4 Rutare între VLAN-uri ... 82
2.5 Rezumat ... 83
2.6 Studiu de caz: ... 84
2.6.1 Comenzi pe switchuri Cisco ... 84
2.6.2 Încapsularea pachetelor: dot1q ... 87
2.7 Realizarea unui bridge între conexiuni în Windows Server 2008 ... 89
3 Adresarea IP ... 92
3.1 Prezentarea protocolului IP ... 92
3.1.1 Structura antetului IPv4 ... 92
3.1.2 Structura antetului IPv6 ... 94
3.1.3 Clase de adrese ... 95
3.1.4 Masca de rețea ... 97
3.1.5 Subrețele ... 98
3.1.6 Super-rețele ... 99
3.1.7 ARP ... 100
3.1.8 DHCP ... 105
3.2 Definirea parametrilor de rețea în Linux ... 106
3.2.1 Configurarea temporară ... 106
3.2.2 Configurarea permanentă ... 107
3.3 Configurarea serviciului DHCP pe un server Linux ... 108
3.3.1 Instalarea și configurarea serverului DHCP ... 108
3.3.2 DHCP Relay ... 110
3.3.3 Exemplu de configurare DHCP ... 110
3.4 Configurarea adreselor de rețea în Windows Server 2008 ... 112
3.4.1 Network and Sharing Center ... 112
3.4.2 Network Connections ... 114
3.4.3 Vizualizarea parametrilor de rețea ... 116
3.4.4 Configurarea manuală a adreselor IP ... 118
3.4.5 Definirea unei configurații IP alternative ... 120
3.4.6 Asignarea automată a adreselor private (APIPA) ... 121
3.5 Configurarea din linia de comandă ... 122
3.5.1 Verificarea unei conexiuni ... 124
3.5.2 Configurări statice și dinamice prin *PowerShell* ... 126
3.6 Studiu de caz ... 128
4 Rutarea în Internet ... 132
4.1 Protocoale de rutare și protocoale rutate ... 132
4.1.1 Ce este Internetul? ... 132
4.1.2 Tabele de rutare ... 133
4.1.3 Clasificarea rutelor ... 134
4.1.4 Rute statice ... 136
4.2 Protocoale rutate ... 136
4.3 Protocoale de rutare ... 137
4.3.1 Determinarea căii optime ... 137
4.3.2 Clasificarea protocoalelor de rutare ... 138
4.3.3 Protocoale distance-vector ... 138
4.3.4 Protocoale link state ... 139
4.4 Sisteme autonome ... 139
4.4.1 Ce este un sistem autonom? ... 139
4.4.2 Protocoale de rutare inter-AS ... 140

Material cu drept de autor

4.5 Configurări la nivel de router în Linux 141
4.5.1 Activarea rutării 141
4.5.2 Configurarea rutelor 142
4.6 NAT - Network Address Translation 144
4.6.1 Translatarea de adrese în Linux 146
4.6.2 Alterarea avansată a pachetelor 147
4.6.3 Tunelare 148
4.6.4 Configurarea tunelului GRE în Linux 149
4.7 Rutarea în Windows Server 2008 150
4.7.1 Routing and remote access services 150
4.8 Studii de caz 157
4.8.1 Încapsularea pachetelor: exemplificare port forwarding 157
4.8.2 Încapsularea pachetelor: exemplu de tunelare 158
5 Wireless 162
5.1 Introducere în rețele wireless 162
5.1.1 Introducere în comunicarea *wireless* 162
5.1.2 Considerente de nivel fizic 163
5.1.3 Standarde pentru rețele locale (WLANs) 165
5.1.4 Wireless MAN 167
5.1.5 Implementarea rețelelor *wireless* 167
5.1.6 Comunicarea *wireless* 170
5.1.7 Securitatea *wireless* 175
5.2 Configurarea unei rețele wireless în Linux – configurări de bază 176
5.2.1 De ce *wireless* pe Linux? 176
5.2.2 Configurări de bază 177
5.3 Configurarea unei rețele wireless în Linux - configurări avansate 184
5.3.1 Partajarea unei conexiuni la Internet într-o rețea *ad hoc* 184
5.3.2 Configurări de securitate în *wireless* 186
5.4 Wireless în Windows Server 2008 191
5.4.1 Activarea serviciului *Wireless* în *Windows Server 2008* 191
5.4.2 Configurarea profilurilor *wireless* 191
5.4.3 Conectarea la o rețea *wireless* 192
5.4.4 Managementul conexiunilor *wireless* 195
5.4.5 Conexiuni *wireless ad hoc* 196
5.5 Administrarea în linie de comandă și PowerShell 198
5.5.1 Managementul serviciului *wireless* prin netsh wlan 198
5.5.2 Managementul serviciului *wireless* prin *PowerShell* 202
6 Securitate și monitorizare 206
6.1 Secure Shell (SSH) 206
6.1.1 Protocolul SSH 207
6.1.2 Configurații de bază SSH 209
6.1.3 Configurații avansate SSH 212
6.2 Firewall 215
6.2.1 Filtrarea de pachete 216
6.2.2 Translatarea de adrese 218
6.2.3 Configurări avansate iptables 220
6.3 Capturare pachetelor și analiza pachetelor. IDS/IPS. 221
6.3.1 Wireshark – configurări de bază 221
6.3.2 Wireshark – configurări avansate 224
6.3.3 Snort – captură de pachete în linie de comandă. IDS/IPS. 225
6.4 Securitate și monitorizare în Windows Server 2008 230
6.4.1 *Windows Firewall* with Advanced Security 230
6.4.2 Monitorizare 241

Material cu drept de autor

7 DNS 245
7.1 Protocolul DNS 245
7.1.1 Domenii DNS 245
7.1.2 Tipuri de servere DNS 247
7.1.3 Tratarea unei cereri DNS 249
7.1.4 Structura bazei de date DNS 251
7.2 Configurări de bază DNS 253
7.2.1 Configurarea clientului DNS pe Linux 253
7.2.2 Utilitare de interogare DNS 253
7.2.3 Configurarea serverului DNS – BIND9 254
7.3 Configurări avansate DNS 264
7.3.1 Delegarea unui subdomeniu DNS. 264
7.3.2 Efectuarea DNS load-balancing. 265
7.3.3 Configurarea DNS Master/Slave 266
7.4 Configurarea unui server DNS pe Windows Server 2008 267
7.4.1 Instalare şi configurare 268
7.5 Configurări în linie de comandă 278
7.5.1 Fişierul Hosts 278
7.5.2 Ipconfig 278
7.5.3 Dnscmd 279
7.5.4 Nslookup 280
8 E-mail 283
8.1 Arhitectură şi funcţionare. Protocoale 284
8.1.1 Funcţionarea serviciului de e-mail 284
8.1.2 Formatul mesajelor 286
8.1.3 SMTP (Simple Mail Transfer Protocol) 287
8.1.4 POP3 (Post Office Protocol) 288
8.1.5 IMAP (Internet Message Access Protocol) 289
8.2 Formatul căsuţei poştale 289
8.2.1 mbox 289
8.2.2 Maildir 290
8.3 Clienţi de e-mail 290
8.3.1 mailx 291
8.4 MTA 291
8.5 Postfix 292
8.5.1 Arhitectură 292
8.5.2 Instalare 292
8.5.3 Interacţiunea cu Postfix 293
8.5.4 Fişiere de configurare 293
8.5.5 Configurare de bază 294
8.5.6 Configurare utilizatori virtuali şi căsuţe poştale virtuale 296
8.5.7 Configurare suport SASL şi TLS 299
8.6 MDA 300
8.6.1 Procmail 300
8.7 Servere de IMAP 303
8.7.1 Courier IMAP Server 303
8.8 Webmail 305
8.9 Studii de caz 305
8.9.1 Comenzi SMTP. Transmiterea unui mesaj folosind SMTP 305
8.9.2 Comenzi POP3. Citirea unui mesaj folosind POP3 306
9 World Wide Web 310
9.1 Modul de funcţionare a Web-ului 310
9.1.1 Uniform Resource Locator (URL) 311

Material cu drept de autor

9.1.2 Hypertext Transfer Protocol ... 311
9.1.3 HyperText Markup Language ... 312
9.1.4 Clienţi web ... 313
9.1.5 Servere web ... 313
9.2 Apache HTTP Server ... 314
9.2.1 Instalare ... 315
9.2.2 Interacţiunea cu serverul web ... 315
9.2.3 Configurare globală ... 316
9.2.4 Găzduire virtuală ... 328
9.3 Configurare suport SSL pentru Apache ... 331
9.3.1 Activare modul. Configurare Port ... 331
9.3.2 Generare certificat ... 332
9.3.3 Configurare virtual host ... 332
9.4 IIS 7 şi Windows Server 2008 ... 333
9.4.1 Avantajele lui IIS 7 ... 334
9.4.2 Instalarea IIS 7 ... 334
9.4.3 Interfaţa de administrare ... 337
9.4.4 Adăugarea unui site web ... 339
9.4.5 Configurarea site-urilor ... 341
9.4.6 Securitate ... 345
9.4.7 Crearea şi întreţinerea jurnalelor ... 348
9.4.8 Crearea de directoare virtuale ... 349
9.4.9 Aplicaţie: Integrarea IIS 7 şi PHP ... 350
9.5 IIS 7 – Configurări în linie de comandă ... 351
10 Securitatea reţelei ... 355
10.1 Riscuri de securitate ... 355
10.1.1 Principii de bază ... 355
10.1.2 Tipuri de atacuri de reţea ... 356
10.1.3 Prevenirea atacurilor ... 364
10.2 Auditarea reţelei ... 365
10.3 Utilitare pentru asigurarea securităţii ... 368
10.3.1 Nmap ... 368
10.3.2 Metasploit ... 370
10.4 Studii de caz ... 373
10.4.1 ARP Poisoning ... 373
10.4.2 Firewalking ... 377
11 Protocoale de nivel transport ... 380
11.1 Noţiuni generale ... 380
11.2 UDP ... 381
11.2.1 Caracteristici UDP ... 381
11.2.2 Formatul datagramelor UDP ... 381
11.3 TCP ... 382
11.3.1 Caracteristici TCP ... 382
11.3.2 Formatul segmentelor TCP ... 383
11.3.3 Conexiunea şi comunicaţia TCP ... 384
11.3.4 Controlul fluxului ... 392
11.4 Studiu de caz: Fragmentarea pachetelor UDP ... 395
12 Anexe ... 399
12.1 Anexa 1: Instalare Windows Server 2008 ... 399
12.2 Anexa 2: Sistemul de boot în Windows Server 2008 ... 404

Material cu drept de autor

0 Introducere în sisteme de operare

"One of the main advantages of Unix over, say, VMS, is the tremendous number of features Unix lacks."

Chris Torek

Ce se învață din acest capitol?

- Noțiunile de bază în administrarea unui sistem de operare
- Administrarea unui sistem Linux
- Noțiuni de bază în *shell scripting*
- Administrarea unui sistem Windows
- *Windows PowerShell*

Cine este...

Linus Torvalds *este programator finlandez, cunoscut cel mai bine ca arhitectul șef al nucleului Linux (BDFL – Benevolent Dictator For Life). După ce a primit o copie a sistemului MINIX, a început lucrul la scrierea Linux pentru i386. S-a mutat in Statele Unite unde susține mișcarea Open Software prin intermediul Linux Foundation. Este angajat al Open Source Development Labs. Dezvoltă în continuare kernelul Linux în cadrul comunității Linux.*

Alan Cox *este un programator britanic implicat în dezvoltarea nucleul Linux. În timp ce era angajat la Universitatea Swansea din Țara Galilor a instalat o distribuție de Linux într-o rețea intens folosită. A început să rezolve numeroase bug-uri și a rescris aproape integral partea de rețea din kernel. A întreținut versiunea 2.2 de Linux, apoi a dezvoltat propria versiune 2.4. A fost pentru o multă vreme considerat în cadrul comunității Linux secundul lui Linus Torvalds. Este un puternic susținător înrăit programelor free/open-source.*

David Miller *este un dezvoltator al nucleului Linux implicat la partea de networking și SPARC. A portat Linux pe arhitectura Sun Microsystem SPARC, argumentând de ce Linux merge mai bine decât Solaris. Este dezvolatorul stivei TCP/IP din Linux și unul din principalii contribuitori la îmbunătățirea performațelor Linux în rețelele cu trafic intens.*

Dave Cutler *este designer și devoltator al sistemelor de operare de la DEC (RSX-11M, VMS, VAXELN) și de la Microsoft (Windows NT). S-a mutat de la DEC la Microsoft pentru a conduce dezvoltarea Windows NT concentrându-se pe implementarea sistemului de operare pe procesorul pe 64biți Alpha de la DEC. Dupa dispariția DEC, a lucrat la portarea Windows pe AMD 64.*

0.1 Ce este un sistem de operare?

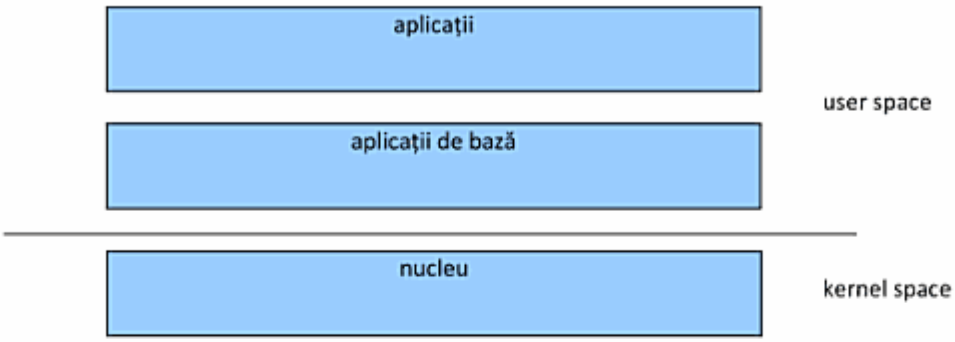
Un sistem de operare este definit de obicei ca un set de programe care facilitează accesul utilizatorului la resursele sistemului. Din punct de vedere conceptual sistemul de operare este văzut ca o **abstractizare** sau ca o **extensie** a mașinii fizice.

Componentele principale ale unui sistem de operare complet, așa cum este el văzut de utilizator, sunt prezentate în figură. Cele trei componente sunt:

- **aplicațiile**: programe folosite direct de utilizator pentru rezolvarea unor sarcini specifice; în această categorie intră suita *Office*, *browser*-e, clienți de e-mail, aplicații multimedia, medii de dezvoltare integrate etc.
- **aplicații de bază**: programele folosite în principal pentru gestiunea și administrarea sistemului sau pentru a asigura servii aplicațiilor de nivel înalt; în această categorie intră interpretorul de comenzi, compilatoare, *linker*-e, biblioteci etc.

Material cu drept de autor

- **nucleul** sau *kernel*-ul: componenta de bază (inima) sistemului de operare; conţine cod care va fi rulat în nivelul privilegiat al procesorului (supervisor) cu scopul de intermediere a accesului la resursele fizice ale sistemului şi de gestiune a acestora; nucleul este componenta esenţială care stabileşte nivelul de performanţă şi de securitate a sistemului de operare.



0-1: Structura unui sistem de operare

Deşi în lumea sistemelor de operare există o mare diversitate, câteva sisteme de operare au o cotă de piaţă şi de utilizare relevantă. Din punct de vedere al destinaţiei, sistemele de operare moderne se împart în sisteme de operare *desktop* (*Windows XP*, *Ubuntu*, *Fedora*, *Xandros*, *Mac OS X*), sisteme de operare server (*Windows 2003 Server*, *Windows 2008 Server*, *Ubuntu Server*, *RedHat Enterprise Linux*) şi sisteme de operare *embedded* (*Windows CE*, *Windows Mobile*, *Symbian*, *Linux*).

Familiile de sisteme de operare cu o cotă semnificativă în piaţă sunt familia Windows, familia Mac OS, familia GNU/Linux şi familia BSD. Sistemul de operare cu cea mai mare cotă pe piaţa dispozitivelor integrate este Symbian.

0.1.1 Ce înseamnă administrarea unui sistem de operare?

Administrarea unui sistem de operare sau *system administration* se referă la activităţile de instalare, întreţinere şi suport pentru sisteme de calcul (de obicei servere) şi configurarea serviciilor pe care aceste sisteme le oferă. Un administrator de sistem este în general o persoană ce deţine un spectru larg de cunoştinţe tehnice şi abilităţi de organizare şi supervizare a diverselor activităţi asociate.

Un administrator de sistem nu este un programator sau inginer *software*. Deşi un administrator nu proiectează sau implementează, de obicei, aplicaţii noi, înţelegerea diverselor programe este necesară. De asemenea, anumite limbaje de programare sunt folosite pentru automatizarea sarcinilor comune. O cerinţă importantă este înţelegerea şi implementarea soluţiilor de securitate legate de buna funcţionare a sistemului.

În general, un administrator de sistem are cunoştinte aprofundate de *scripting* care îi permit automatizarea sarcinilor şi obţinerea periodică de informaţii.

Diverse organizaţii oferă training şi examene de certificare pentru administratorii de sistem pentru diverse sisteme de operare: MCSA (*Microsoft Certified System Engineer*), RHCE (*RedHat Certified Engineer*), SCNA (*Sun Certified Network Administrator*).

0.2 Noţiuni de bază pentru administrarea unui sistem de operare Linux

Cunoştinţele şi activităţile necesare pentru adminstrarea unui sistem de operare Linux sunt similare cu cele necesare pentru orice sistem Unix. Componentele importante sunt gestiunea dispozitivelor *hardware*, sistemului de fişiere, gestiunea utilizatorilor, gestiunea

Material cu drept de autor

pachetelor de programe, gestiunea serviciilor, asigurarea securității sistemului, automatizarea sarcinilor.

Această carte se va referi cu predilecție la gestiunea serviciilor. Vor fi prezentate și informații utile despre alte componente necesare.

Majoritatea interacțiunii administratorului de sistem cu sistemul de operare Linux se va realiza prin intermediul interfeței în linia de comandă (*shell*) și a fișierelor de configurare text (cu ajutorul unui editor). Se vor considera acoperite cunoștințele de bază despre utilizarea interfeței în linia de comandă și editarea fișierelor de configurare.

0.2.1 Componentele unui sistem GNU/Linux. Distribuții

Un sistem de operare GNU/Linux este compus din nucleul (*kernel*-ul) Linux și aplicațiile ce rulează peste acesta. O bună parte din aceste aplicații sunt parte din proiectul GNU[1].

Una dintre cele mai importante aplicații este interpretorul de comenzi (*shell*-ul). *Shell*-ul implicit pe majoritatea distribuțiilor Linux este *Bash*. *Shell*-ul acționează ca un intermediar între utilizator și nucleu. *Shell*-ul transformă comenzile introduse de utilizator în procese care folosesc nucleul pentru realizarea unei sarcini.

Alte aplicații de bază importante sunt editoare, compilatoare, biblioteci. În general aplicațiile grafice lipsesc de pe un sistem server, interacțiunea realizându-se aproape exclusiv prin intermediul interfeței în linie de comandă.

Spre deosebire de multe alte sisteme de operare, dezvoltarea nucleului și a aplicațiilor se realizează diferit. Agregarea acestor componente se realizează prin intermediul unei distribuții GNU/Linux. Există sute de distribuții Linux, printre cele mai cunoscute numărându-se Ubuntu, Fedora/RedHat, SuSE, Debian, Gentoo, Slackware etc.

Unele distribuții sunt similare. Un astfel de exemplu sunt distribuțiile Debian-based: Debian, Ubuntu, MEPIS, Damn Small Linux, Xandros, Linspire. Aceste distribuții folosesc pachetele software puse la dispoziție de proiectul Debian (actualmente în număr de peste 26000) și sistemul APT[2] de gestiune a pachetelor.

0.2.2 Sistemul de fișiere

Nucleul Linux oferă suport pentru un număr impresionat de sisteme de fișiere. Cu toate acestea, interfața oferită utilizatorului este aceeași indiferent de tipul sistemului de fișiere din spate. În general, denumirile diverselor fișiere și directoare sunt simple pentru a putea fi folosite eficient din linia de comandă (`/bin`, `/var`, `/usr`, `/lib`). În opoziție, Mac OS X folosește denumiri mai clare (`/Library/`, `/Applications/`, `/Users/`). Separatorul folosit este / (*slash*).

Majoritatea distribuțiilor Linux oferă o interfață compatibilă cu *Filesystem Hierarchy Standard*[3]. FHS definește numele directoarelor principale și a conținutului acestora într-o distribuție Linux. Câteva din intrările importante sunt precizate în tabelul de mai jos:

Director	Descriere
/	Rădăcina sistemului de fișiere
`/bin/`	Executabile (binare) asociate comenzilor importante
`/dev/`	Dispozitive (/dev/null, /dev/hda, /dev/random)
`/etc/`	Fișiere de configurare

1 http://www.gnu.org/
2 http://www.debian.org/doc/manuals/apt-howto/
3 http://www.pathname.com/fhs/

Material cu drept de autor

/home/	Directoarele *home* ale utilizatorilor
/lib/	Biblioteci
/mnt/	Sisteme de fișiere montate temporar
/proc/	Sistemul de fișiere procfs
/root/	*Home*-ul utilizatorului privilegiat (*root*)
/sbin/	Executabilele comenzilor ce necesită drepturi de utilizator privilegiat
/usr/	Ierarhie secundară: conține binare (/usr/bin), biblioteci (/usr/lib)
/var/	Fișiere variabile (jurnale, cozi, temporare)
/var/log/	Fișiere de jurnalizare pentru diverse aplicații

Comenzile de bază pentru interacțiunea cu sistemul de fișiere sunt: `pwd`, `ls`, `cd`, `touch`, `rm`, `mkdir`, `rmdir`, `cp`, `mv`, `link`, `unlink`.

0.2.3 Gestiunea utilizatorilor

Gestiunea utilizatorilor se referă la adăugarea de noi utilizatori, ștergerea unui utilizator existent, modificarea informațiilor despre un utilizator și afișarea diverselor informații.

În sistemele Unix, informațiile despre utilizatori sunt reținute în fișierul `/etc/passwd`. Fiecare linie din acest fișier conține numele utilizatorului, identificatorului său, *home*-ul, *shell*-ul rulat în momentul autentificării și alte informații de descriere:

```
anaconda:~# cat /etc/passwd
andreir:x:1114:1026:Andrei Rizoiu:/home/students/andreir:/bin/bash
alexn:x:1115:1026:Alex Negrea:/home/students/alexn:/bin/bash
[...]
```

Din motive de securitate, *hash*-ul asociat parolei nu se găsește în fișierul `/etc/passwd`, ci în fișierul `/etc/shadow` care nu poate fi accesat de majoritatea utilizatorilor:

```
anaconda:~# ls -l  /etc/shadow
-rw-r----- 1 root shadow 7068 2008-09-12 11:59 /etc/shadow
```

Pentru a afla informații despre un utilizator al sistemului se pot folosi comnzile `id` sau `finger`:

```
anaconda:~# id andreir
uid=1114(andreir) gid=1026(students) groups=1026(students),1037(rl)
anaconda:~# finger alexn
Login: alexn                            Name: Alex Negrea
Directory: /home/students/alexn         Shell: /bin/bash
Never logged in.
No mail.
No Plan.
```

Utilizatorul privilegiat într-un sistem Unix este utilizatorul `root` cu `uid`-ul 0 și *home*-ul în `/root`:

```
anaconda:~# head -1 /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

Utilizatorul `root` (de fapt utilizatorul cu `uid`-ul 0) are drepturi absolute în cadrul sistemului și poate rula orice comandă. Se recomandă folosirea unui cont neprivilegiat. Doar atunci când este nevoie se va folosi contul privilegiat.

Schimbarea unui utilizator se realizează cu ajutorul comenzii `su` urmată de introducerea parolei pentru acel utilizator. Dacă utilizatorul inițial este `root`, nu se solicită introducerea parolei:

```
anaconda:~# head -1 /etc/passwd
```

Material cu drept de autor

```
root:x:0:0:root:/root:/bin/bash
anaconda:~# su - andreir
andreir@anaconda:~$ su - razvan
Password:
```

Sistemele Ubuntu dezactivează, de obicei, utilizatorul `root` și recomandă folosirea comenzii `sudo`. Comanda `sudo`, împreună cu fișierul `/etc/sudoers` permite rularea de comenzi privilegiate de către un utilizator neprivilegiat. De obicei, un utilizator neprivilegiat care are drept de `sudo` va rula comanda `sudo bash` pentru a obține un *shell* cu drepturi privilegiate.

Schimbarea parolei unui utilizator se realizează cu ajutorul comenzii `passwd`. Utilizatorul privilegiat poate schimba parola oricărui utilizator. Un utilizator neprivilegiat își poate schimba parola doar sieși:

```
anaconda:~# passwd guest
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

În sistemele Debian-based, adăugarea, respectiv ștergerea unui utilizator se realizează prin intermediul scripturilor `adduser` și `deluser`:

```
anaconda:~# adduser test
Adding user `test' ...
Adding new group `test' (1038) ...
Adding new user `test' (1003) with group `test' ...
Creating home directory `/home/test' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for test
Enter the new value, or press ENTER for the default
        Full Name []: Test User
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [y/N] y
anaconda:~# id test
uid=1003(test) gid=1038(test) groups=1038(test)
anaconda:~# deluser --remove-home test
Looking for files to backup/remove ...
Removing files ...
Removing user `test' ...
Done.
```

Avantajul și, în același timp, dezavantajul folosirii comenzii `adduser` este interactivitatea. Automatizarea sarcinilor presupune comenzi non-interactive. Pentru aceasta, se pot folosi comenzile `useradd`, `userdel` și `usermod`. `useradd`, respectiv `userdel` sunt folosite de scripturile `adduser` și `deluser`.

```
anaconda:~# useradd -m -d /home/test test
anaconda:~# id test
uid=1116(test) gid=1116(test) groups=1116(test)
anaconda:~# usermod -s /bin/sh test
anaconda:~# userdel -r test
anaconda:~# id test
id: test: No such user
```

0.2.4 Drepturi pe sistemul de fișiere

Tradițional, sistemele Unix folosesc un sistem simplificat de asociere a drepturilor pe o intrare în sistemul de fișiere. Orice fișier este deținut de un utilizator și un grup. Există astfel trei categorii de utilizatori: utilizatorul care deține fișierul (*user*), grupul care deține fișierul (*group*), ceilalți utilizatori (*others*). Prescurtat cele trei categorii sunt denumite `ugo`.

Fiecare dintre cele trei categorii are trei drepturi posibile: citire (*read*), scriere (*write*) și execuție (*execute*). Prescurtat cele trei drepturi sunt denumite `rwx`. Fiecare din cele trei categorii de utilizatori poate avea oricare și oricâte din cele trei drepturi și sunt exprimate de obicei într-o formă liniară de comanda ls:

Material cu drept de autor

```
anaconda:~# ls -l /etc/services
-rw-r--r-- 1 root root 18274 2007-02-02 04:09 /etc/services
anaconda:~# ls -l /var/mail/razvan
-rw-rw---- 1 razvan mail 0 2007-06-19 16:54 /var/mail/razvan
```

Există și un echivalent binar al drepturilor exprimat în octal. Astfel, cele două fișiere de mai sus au drepturile 644, respectiv 660 în octal.

Cele trei drepturi de mai sus au semnificații diferite când sunt folosite peste fișiere sau directoare, conform tabelului de mai jos:

Drept	Efect fișier	Efect director
read	Fișierul poate fi vizualizat (`cat`, `less`)	Poate fi vizualizat conținutul său (`ls`)
write	Fișierul poate fi scris (un editor) sau șters (`rm`)	Pot fi create/șterse noi intrări (`touch`, `rm`, `mkdir`, `rmdir`)
execute	Fișierul (binar sau script) poate fi executat	Directorul poate fi parcurs (poate fi parte a unei căi)

0-2: Drepturile directoarelor și fișierelor

Schimbarea drepturilor pe un fișier se realizează cu ajutorul comenzii `chmod`. Comanda are efect doar dacă este rulată de utilizatorul ce deține fișierul:

```
razvan@anaconda:/tmp$ touch a.txt
razvan@anaconda:/tmp$ ls -l a.txt
-rw-r--r-- 1 razvan razvan 0 Sep 12 17:47 a.txt
razvan@anaconda:/tmp$ chmod a+w a.txt
razvan@anaconda:/tmp$ ls -l a.txt
-rw-rw-rw- 1 razvan razvan 0 Sep 12 17:47 a.txt
razvan@anaconda:/tmp$ chmod u+x,g-r,o-w a.txt
razvan@anaconda:/tmp$ ls -l a.txt
-rwx-w-r-- 1 razvan razvan 0 Sep 12 17:47 a.txt
razvan@anaconda:/tmp$ chmod 744 a.txt
razvan@anaconda:/tmp$ ls -l a.txt
-rwxr--r-- 1 razvan razvan 0 Sep 12 17:47 a.txt
```

Schimbarea deținătorului și grupului ce deține fișierul se realizează cu ajutorul comenzii `chown`. Comanda `chown` poate fi rulată doar de utilizatorul privilegiat:

```
anaconda:/tmp# ls -l a.txt
-rwxr--r-- 1 razvan razvan 0 2008-09-12 17:47 a.txt
anaconda:/tmp# chown guest:projects a.txt
anaconda:/tmp# ls -l a.txt
-rwxr--r-- 1 guest projects 0 2008-09-12 17:47 a.txt
anaconda:/tmp# chown mircea a.txt
anaconda:/tmp# ls -l a.txt
-rwxr--r-- 1 mircea projects 0 2008-09-12 17:47 a.txt
```

0.2.5 Gestiunea pachetelor

Un pachet, sau un pachet *software*, este o aplicație sau o componentă accesibilă în forma unei arhive care poate fi instalată de un sistem de gestiune a pachetelor (PMS – *Package Management System*). De obicei, pachetele reprezintă programe precompilate care pot fi instalate ușor, spre deosebire de instalarea din surse care este mai anevoioasă.

Lucrul cu pachete (instalare, dezinstalare, configurare) se realizează prin intermediul unui sistem de gestiune a pachetelor (precum `dpkg`, `rpm`, `pacman`). Un astfel de pachet conține, în afară de fișierele asociate programului și un set de metainformații precum versiunea pachetului, descrierea și dependințe. PMS-ul folosește aceste informații pentru a decide dacă se va realiza instalarea pachetului, *upgrade*-ul acestuia, instalarea dependințelor etc. O dependință între pachetele A și B înseamnă că instalarea pachetului A necesită instalarea pachetului B. La fel, dezinstalarea pachetului B va forța dezinstalarea pachetului A.

Material cu drept de autor

Majoritatea distribuțiilor GNU/Linux folosesc noțiunea de depozit de pachete (*repository*). Acesta este un URL care precizează locația diverselor pachete ale distribuției. Aceste depozite sunt precizate în fișiere de configurare specifice distribuției. Aplicații *front-end* peste PMS pot interoga depozitele și pot descărca și instala noi pachete.

În lumea Linux există diverse formate de pachete, cele mai cunoscute fiind formatul DEB, specific distribuțiilor Debian-based și formatul RPM folosit de Fedora/RedHat, Mandriva, SuSE, etc. Fiecare format are propriul PMS. Utilitarul `alien`[1] permite conversia între diverse formate de pachete.

0.2.6 Gestiunea pachetelor DEB

Utilitarul de bază (PMS) pentru gestiunea pachetelor DEB este `dpkg`. `dpkg` este folosit pentru instalarea, dezinstalarea și configurarea unui pachet.

În mod obișnuit, însă, cea mai mare parte a acestor acțiuni vor fi realizate prin intermediul utilitarului APT (*Advanced Packaging Tool*). APT este un *front-end* peste `dpkg` și permite interogarea depozitelor de pachete configurate, verificarea dependințelor, descărcarea automată a pachetelor din *repository*, actualizarea acestora, *upgrade*-ul unei distribuții, etc.

Fișierul de configurare a unui depozit DEB este `/etc/apt/sources.list`:

```
anaconda:/tmp# cat /etc/apt/sources.list
[...]
deb http://ftp.lug.ro/debian etch main contrib non-free
deb-src http://ftp.lug.ro/debian etch main contrib non-free
```

Adăugarea unui nou depozit înseamnă adăugarea unei noi linii în fișierul de configurare.

Acțiunile care pot fi realizate cu ajutorul utilitarului `apt` sunt:

- actualizarea listei de pachete:

```
anaconda:/tmp# apt-get update
Get: 1 http://ftp.lug.ro etch Release.gpg [386B]
Get: 2 http://ftp.lug.ro etch/updates Release.gpg [189B]
Hit http://ftp.lug.ro etch Release
Get: 3 http://www.backports.org etch-backports Release.gpg [189B]
Get: 4 http://ftp.lug.ro etch/updates Release [37.6kB]
Ign http://debian.pkgs.cpan.org unstable Release.gpg
Get: 5 http://www.backports.org etch-backports Release [43.7kB]
Ign http://ftp.lug.ro etch/main Packages/DiffIndex
Ign http://ftp.lug.ro etch/contrib Packages/DiffIndex
[...]
```

- căutarea de pachete:

```
anaconda:/tmp# apt-cache search hevea
hevea - translates from LaTeX to HTML, info, or text
lyx - High Level Word Processor
hevea-doc - HeVeA documentation
```

- afișarea de informații despre un fișier:

```
anaconda:/tmp# apt-cache show hevea
Package: hevea
Priority: optional
Section: tex
Installed-Size: 2125
Maintainer: Debian OCaml Maintainers <debian-ocaml-maint@lists.debian.org>
Architecture: all
Version: 1.09-3
```

- instalarea unui pachet (și a dependințelor sale):

```
anaconda:/tmp# apt-get install apt-file
Reading package lists... Done
Building dependency tree... Done
The following extra packages will be installed:
  libapt-pkg-perl libconfig-file-perl
The following NEW packages will be installed
  apt-file libapt-pkg-perl libconfig-file-perl
0 upgraded, 3 newly installed, 0 to remove and 66 not upgraded.
```

[1] http://kitenet.net/~joey/code/alien/

Material cu drept de autor

```
Need to get 106kB of archives.
After unpacking 406kB of additional disk space will be used.
Do you want to continue [Y/n]? Y
```

- dezinstalarea unui pachet:

```
anaconda:/tmp# apt-get remove --purge apt-file
Reading package lists... Done
Building dependency tree... Done
The following packages will be REMOVED
  apt-file*
```

- curățarea cache-ului local de pachete:

```
anaconda:/tmp# apt-get clean
```

- instalarea surselor unui pachet:

```
anaconda:/tmp# apt-get source apt-file
Reading package lists... Done
Building dependency tree... Done
Need to get 17.7kB of source archives.
Get: 1 http://ftp.lug.ro etch/main apt-file 2.0.8.2 (dsc) [505B]
Get: 2 http://ftp.lug.ro etch/main apt-file 2.0.8.2 (tar) [17.2kB]
```

În plus față de `apt`, `dpkg` oferă opțiuni pentru interogarea stării actuale a pachetelor sau a conținutul acestora. Printre opțiunile utile se numără:

- listarea conținutului unui pachet:

```
anaconda:/tmp# dpkg -L coreutils
/.
/bin
/bin/mkdir
/bin/mv
/bin/true
/bin/mknod
/bin/sleep
/bin/touch
/bin/chgrp
/bin/uname
/bin/echo
/bin/sync
[...]
```

- afișarea pachetelor al căror nume se potrivește cu o expresie regulată:

```
anaconda:/tmp# dpkg -l 'linux*'
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Installed/Config-files/Unpacked/Failed-config/Half-installed
|/ Err?=(none)/Hold/Reinst-required/X=both-problems (Status,Err: uppercase=bad)
||/ Name           Version        Description
+++-==============-==============-============================================
un  linux          <none>         (no description available)
un  linux-doc-2.6. <none>         (no description available)
un  linux-gnu      <none>         (no description available)
un  linux-image    <none>         (no description available)
un  linux-image-2. <none>         (no description available)
ii  linux-image-2. 2.6.18+6etch3  Linux kernel 2.6 image on Ppro/Celeron/PII/P
[...]
```

- căutarea pachetului ce conține un anumit fișier

```
anaconda:/tmp# dpkg -S /bin/ps
procps: /bin/ps
```

0.2.7 Gestiunea serviciilor

O parte importantă a acestei cărți este dedicată diverselor servicii pe care un sistem de operare le pune la dispoziție (web, DNS, e-mail). În majoritate, serviciile de rețea pe care un sistem Linux le oferă au o formă de administrare comună: instalare, fișiere de configurare, pornire, repornire, oprire, *troubleshooting*, jurnalizare.

Un serviciu de rețea (web, DNS, e-mail) este implementat printr-un proces server. Un proces server este asociat, în lumea Unix, cu un *daemon*. Un *daemon* este un proces care rulează în *background* decuplat de orice terminal care de obicei ascultă conexiuni pe un

Material cu drept de autor

anumit port şi oferă resurse sau informaţii unui client. Exemple de astfel de *servere/daemoni* sunt: `bind` (*Berkeley Internet Name Daemon*), *Apache*/`httpd`, `postfix`, `sshd`, `courier-imap`.

În general, un serviciu este pornit la iniţializarea sistemului de procesul `init`, primul proces pornit de sistemul de operare. De aceea, în general, un *daemon* va avea asociat un script de interacţiune în `/etc/init.d/`. Operaţiile de pornire, oprire, repornire a unui *daemon* pot fi realizate, în mod generic, cu ajutorul unui astfel de script:

```
anaconda:/tmp# /etc/init.d/apache
Usage: /etc/init.d/apache {start|stop|reload|reload-modules|force-reload|restart}
anaconda:/tmp# /etc/init.d/bind9
Usage: /etc/init.d/bind9 {start|stop|reload|restart|force-reload}.
anaconda:/tmp# /etc/init.d/postfix
Usage: /etc/init.d/postfix {start|stop|restart|reload|flush|check|abort|force-reload}.
anaconda:/tmp# /etc/init.d/courier-imap
Usage: /etc/init.d/courier-imap {start|stop|restart|reload|force-reload}
```

Fiecare *server/daemon* poate avea şi un mecanism propriu de pornire/repornire (`apache2ctl` pentru Apache sau `postfix` pentru *Postfix*) dar interfaţa `/etc/init.d/` este generică şi comună oricărui *daemon*.

Pentru a verifica faptul că un *daemon* rulează, se poate folosi comanda `netstat` pentru a afişa *daemon*-ii care ascultă conexiunii în reţea:

```
anaconda:/tmp# netstat --tcp --listening --numeric --programs
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program
name
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      3285/apache
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN      3285/apache
tcp        0      0 0.0.0.0:113             0.0.0.0:*               LISTEN      3080/inetd
tcp        0      0 0.0.0.0:21              0.0.0.0:*               LISTEN      3179/vsftpd
tcp        0      0 141.85.37.25:53         0.0.0.0:*               LISTEN      2779/named
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      2779/named
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN      3142/master
[...]
```

Fiind detaşate de un terminal de control, interacţiunea cu serviicile se realizează prin intermediul fişierelor de configurare. Se modifică fişierul/fişierele de configurare asociate unui anumit serviciu şi se reporneşte serviciul pentru a forţa recitirea acestora. Fişierele de configurare pentru servicii sunt fişiere text, sunt localizate în `/etc` şi conţin de obicei directive de configurare în forma „`nume valoare`”:

```
anaconda:/tmp# cat /etc/postfix/main.cf
[...]
myhostname = anaconda.cs.pub.ro
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
myorigin = /etc/mailname
[...]
```

Diversele probleme care apar în cazul unei configurări invalide, defectuoase sau a altor probleme sunt identificate prin intermediul jurnalelor. Fiecare serviciu foloseşte un fişier sau subdirector în `/var/log` unde stochează diverse mesaje de informare sau avertizare pentru administratorul de sistem. O intrare în cadrul fişierului de jurnalizare indică ora la care s-a realizat intrarea (*timestamp*) şi un mesaj de informare. Se foloseşte, de obicei, utilitarul `tail` pentru a afişa ultimele intrări dintr-un fişier de jurnalizare:

```
anaconda:/tmp# tail /var/log/mail.err
Sep 10 11:15:33 anaconda imapd-ssl: Error reading ACLs for INBOX.lists.gnupg.: Invalid
argument
Sep 10 11:16:04 anaconda last message repeated 5 times
Sep 10 11:16:26 anaconda last message repeated 4 times
Sep 12 12:14:18 anaconda courierpop3login: authentication error: Input/output error
[...]
```

0.2.8 *Shell scripting*

O parte importantă din sarcinile unui administrator de sistem sunt repetitive şi pot fi uşor automatizate. De aceea, cunoştinţele de *shell scripting* sunt fundamentale pentru a asigura

Material cu drept de autor

eficiența activităților efectuate. Un administrator poate recurge și la alte limbaj de *scripting* precum *Perl* sau *Python* dar *shell scripting*-ul oferă posibilitatea de a folosi comenzi deja implementate.

Fără a-și propune să prezinte exhaustiv noțiunile legate de *shell scripting*, această secțiune oferă o trecere în revistă a aspectelor importante.

0.2.8.1 Structura unui script *shell*

Un script *shell* conține pe prima linie simbolul *she-bang* (#!) urmat de interpretorul folosit, spre exemplu `#!/bin/bash`. Următoarele linii sunt instrucțiuni sau comenzi *shell* rulate secvențial. Un exemplu simplu de script *shell* este cel de mai jos:

```
razvan@anaconda:/tmp$ cat hw.bash
#!/bin/bash

echo "Hello, World"

exit 0
```

Rularea acestui script se poate realiza prin transmiterea ca argument interpretorului sau prin rularea acestuia ca un executabil (dacă are drept de execuție):

```
razvan@anaconda:/tmp$ bash hw.bash
Hello, World
razvan@anaconda:/tmp$ ls -l hw.bash
-rw-r--r-- 1 razvan razvan 41 Sep 13 14:45 hw.bash
razvan@anaconda:/tmp$ chmod a+x hw.bash
razvan@anaconda:/tmp$ ./hw.bash
Hello, World
```

0.2.8.2 Redirectare și înlănțuirea comenzilor

Redirectarea înseamnă folosirea unui fișier pentru a reține ieșirea unei comenzi sau pentru a fi folosit ca intrare a unei comenzi. Redictarea ieșirii se realizează folosind operatorul >, iar a intrării folosind <:

```
anaconda:/tmp# ls > out.txt
anaconda:/tmp# cat < out.txt
apt-file-2.0.8.2
apt-file_2.0.8.2.dsc
apt-file_2.0.8.2.tar.gz
[...]
```

Înlănțuirea comenzilor se referă la folosirea operatorului | (*pipe*) pentru a trimite ieșirea unei comenzi la intrarea alteia:

```
razvan@valhalla:~$ last -10 | cut -d ' ' -f 1 | sort -u
razvan
reboot
wtmp
```

0.2.8.3 Variabile

În programarea *shell*, ca și multe alte limbaje de *scripting* o variabilă nu are un tip și poate fi considerată, în funcție de nevoie, șir sau număr. Câteva exemple de inițializare de variabile sunt enumerate mai jos:

```
var1=5
my_home_dir=/home/users/alpha
list="a b c d e"
Referirea unei variabile se realizează prin prefixarea numelui acesteia cu $.
razvan@valhalla:~$ echo $a
5
razvan@valhalla:~$ b="a are valoarea $a"
razvan@valhalla:~$ echo $b
a are valoarea 5
```

Material cu drept de autor

Operatorul `$` poate fi folosit, în mod generic, pentru expandare. Exceptând expandarea unei variabile la valoarea acesteia, operatorul poate fi folosit pentru expandare aritmetică sau pentru stocarea ieşirii unei comenzi:

```
razvan@valhalla:~$ a=5
razvan@valhalla:~$ b=$((a+3))
razvan@valhalla:~$ echo $b
8
razvan@valhalla:~$ c=$(ls | wc -l)
razvan@valhalla:~$ echo $c
15
```

0.2.8.4 Cicluri şi instrucţiuni de decizie

Instrucţiunea de decizie folosită în scripturi este if. If primeşte ca argument o comandă. Rezultatul întors de această comandă determină execuţia sau nu a instrucţiunilor următoare. De obicei se foloseşte comanda test:

```
if test -f $fname; then                    # este $fname un fisier
    [...]
fi
if test $a -ge 5; then                # este $a mai mare sau egal cu 5
    [...]
fi
if test "abc" == $string_var; then    # este $string_var egal cu şirul "abc"
    [...]
fi
```

Cele mai folosite instrucţiuni de ciclare sunt `for` şi `while`. Mai jos sunt prezentate câteva exemple de folosire a acestora:

```
razvan@anaconda:/tmp$ for i in $(seq 1 10); do echo $i; done
1
2
3
4
5
6
7
8
9
10
razvan@anaconda:/tmp$ for i in "a b c"; do touch $i; done
razvan@anaconda:/tmp$ ls -l a b c
-rw-r--r-- 1 razvan razvan 0 Sep 13 14:51 a
-rw-r--r-- 1 razvan razvan 0 Sep 13 14:51 b
-rw-r--r-- 1 razvan razvan 0 Sep 13 14:51 c
razvan@anaconda:/tmp$ i=1; while test $i -le 5; do echo $i; ((i++)); done
1
2
3
4
5
```

0.2.8.5 Filtre de text

O parte importantă a scripturilor *shell* este folosită pentru prelucrarea de informaţii text, fie fişiere text fie ieşirea anumitor comenzi. O componentă importantă a unui script *shell* este reprezentată, în acest context, de comenzi folosite pentru prelucrarea de informaţii text, denumite şi filtre de text. Exemple de astfel de comenzi sunt: `tail`, `head`, `grep`, `cut`.

Comenzile `tail` şi `head` sunt folosite pentru a afişa primele sau ultimele linii dintr-un fişier text. Comanda `grep` este folosită pentru a selecta linii care conţin un anumit şablon. Comanda `cut` selectează coloane:

```
razvan@anaconda:/tmp$ cat /etc/passwd | grep and
andrewbwm:x:1031:1026:Andrei Buhaiu:/home/students/andrewbwm:/bin/bash
gabi:x:1039:1026:Gabriel Sandu:/home/students/gabi:/bin/bash
andreic:x:1045:1026:Andrei Cibotaru:/home/students/andreic:/bin/bash
antand:x:1069:100:,,,:/home/users/antand:/bin/bash
alexj:x:1072:1026:Alexandru Juncu:/home/students/alexj:/bin/bash
andreea:x:1082:1028:Andreea Leta:/home/students/andreea:/bin/bash
[...]
razvan@anaconda:/tmp$ cat /etc/passwd | cut -d ':' -f 1,5
[...]
```

Material cu drept de autor

```
iulian:Iulian Moraru,,,
max:Maximilian Machedon,,,
cosmin:Cosmin Ratiu,,,
adrian:Adrian Nistor,,,
cristina:Cristina Carbunaru,,,
amihaiuc:Alex Mihaiuc,,,
```

0.3 Introducere în Microsoft Windows Server 2008

Alegerea şi instalarea unui NOS (*Network Operating System – un sistem de operare conceput pentru a funcţiona în reţea*) reprezintă una dintre cele mai importante sarcini ce se regăsesc printre responsabilităţile unui administrator de reţea. De capabilităţile sistemelor de operare instalate depind toate procesele ulterioare de administrare, configurare, securizare, actualizare, extindere, *backup* şi, de ce nu, de atingere a unui *user-experience* favorabil.

Aceasta este prima ediţie a cărţii de faţă în care administrarea serviciilor de reţea va fi tratată deopotrivă din perspectiva Linux/UNIX cât şi din punctul de vedere al alternativei Microsoft în domeniul NOS-urilor: *Windows Server 2008*.

A fost ales *Windows Server 2008* pentru această ediţie a cărţii nu pentru a-i pune în lumină plusurile sau minusurile faţă de mediul Linux, ci pentru a prezenta o alternativă demnă de luat în considerare din acest domeniu, o opţiune funcţională şi aplicabilă, complet sau parţial, în conceperea şi implementarea unei reţele de calculatoare.

Fiecare capitol, în limita aplicabilităţii, va prezenta, în plus, tehnici şi opţiuni de configurare a serviciilor de reţea în mediul Windows. Deoarece *Windows Server 2008* reprezintă deja o soluţie proprietară, se va încerca limitarea gradului de acoperire a serviciilor la nivelul celor disponibile implicit într-o instalare de *Windows Server 2008*, cu observaţiile de rigoare acolo unde pot apărea diferenţe legate de versiuni. De asemenea, va fi dedicat un efort substanţial pentru evidenţierea şi utilizarea posibilităţilor de *scripting* oferite de Windows prin *PowerShell*.

0.3.1 Despre Microsoft *Windows Server 2008*

La momentul scrierii acestei cărţi, *Windows Server 2008* reprezintă cel mai recent produs din gama sistemelor de operare de la Microsoft şi, totodată, cel mai recent din seria Server. În paginile acestei cărţi vor fi evidenţiate facilităţile oferite de *Windows Server 2008* într-un context unitar, fără a accentua doar ceea ce este specific lui *Windows Server 2008*. De aceea, este posibil ca multe dintre conceptele, procedurile şi serviciile descrise să corespundă într-o oarecare măsură celor prezente în versiunile anterioare din serie: *Windows Server 2003*, *Windows 2000*, etc, cu *Service Pack*-urile aferente.

Windows Server 2008 intră în scena sistemelor de operare ca un succesor direct al lui *Windows Server 2003*, care s-a bucurat de un succes semnificativ. Înainte de a fi definitivat, în faza sa incipientă, el a fost denumit *Codename Longhorn* care, în cele din urmă, s-a dovedit a fi o platformă de bază atât pentru varianta prezentă de Windows Server cât şi pentru Windows *Vista*, lansat cu ceva timp înaintea versiunii Server. Din cauza acestei baze comune, *Windows Server 2008* şi *Vista* împărtăşesc destul de multe proprietăţi ce ţin de arhitectură şi funcţionalitate de bază.

Atât *Windows Server 2008* cât şi *Vista* oferă facilităţi similare cu privire la securitate, management şi administrare, în general, împreună cu un suport solid pentru IPv6, utilitare *wireless* native precum şi multe alte facilităţi ce se conformează cerinţelor reţelelor, tehnologiilor şi utilizatorilor din prezent. Bineînţeles, distincţia dintre cele două devine drastică în momentul în care *Windows Server 2008* se depărtează de mediul preponderent *desktop/workstation* al lui *Vista* prin opţiunile de implementare la scară largă, în medii *enterprise*, utilitare de monitorizare şi diagnostic, redundanţă şi facilităţi de securitate mult îmbunătăţite.

În sensul său de bază, un „*server*" trebuie să ofere servicii unor utilizatori sau altor servere, ori, ca în cazul cel mai des intâlnit în lumea reală, o combinaţie a celor două. În termeni tehnici, serverul reprezintă, de fapt, sistemul de operare al maşinii ce îndeplineşte rolul de server, împreună cu aplicatiile pe care acesta le susţine şi le foloseşte pentru a putea oferi serviciile menţionate anterior. Evident, în aceste condiţii, o platformă de tip server va trebui să suporte un nivel diferit de încărcare şi un mod diferit de utilizare a resurselor sale faţă de o platformă *desktop/workstation*. De asemenea, un server trebuie să poată funcţiona o perioadă îndelungată de timp fără supraveghere şi să implementeze mecanisme sigure de recuperare în cazul erorilor sau evenimentelor neprevăzute. Totodată, acestea trebuie documentate într-un sistem de menţinere a jurnalelor (*logging*) bine pus la punct pentru a putea oferi rapid şi coerent informaţii despre evoluţia sistemului. Acestea sunt, de asemenea, aspecte prin care *Windows Server 2008* se distanţează de *Vista* şi, implicit, de orice alt sistem de operare orientat spre mediul *desktop*.

0.3.2 *Windows PowerShell*

0.3.2.1 Ce este *Windows PowerShell*?

Pe scurt, *PowerShell* reprezintă noua generaţie de interpretor de comenzi şi limbaj de *scripting* de la Microsoft, construit peste platforma .NET şi CLR (*Common Language Runtime*), ce poate fi folosit atât pentru înlocuirea „venerabilului" interpretor de comenzi `cmd.exe`, cât şi a limbajului VBScript. Această natură „dualã" a lui *PowerShell* poate crea iniţial probleme printre administrator obişnuiţi cu `cmd.exe` şi slabele sale capabilităţi sau cu VBScript, depotrivă puternic şi dificil (incomod) folosit pentru automatizarea sarcinilor. Aceste unelte funcţionează satisfăcător din punctul de vedere al funcţionalităţii pe care o oferă, dar sunt în prezent folosite în scopuri pentru care nu au fost create. `Cmd.exe` a fost scris ca un interpretor succesor al *DOS Prompt*-ului, iar VBScript a fost mai mult sau mai puţin conceput în contextul paginilor web. Niciunul nu a fost conceput de la bun început de către administratori sau pentru ei.

Majoritatea interpretoarelor de comenzi, printre care şi `cmd.exe`, alături de interpretoarele din mediul UNIX (`sh`, `csh`, `ksh`, `bash`, etc) funcţionează prin executarea comenzilor în cadrul unor procese noi, returnând rezultatul sub formă de text utilizatorului. Din acest motiv, de-a lungul timpului, în special în mediul UNIX, utilitarele de procesare de text au fost îmbunătăţite în permanenţă, oferind în mod indirect funcţionalităţi din ce în ce mai extinse interpretorului în sine şi uşurând interacţiunea utilizatorului cu el. De asemenea, interpretoarele acceptă comenzi ce sunt încorporate direct în funcţionalitatea lor, la această categorie pretându-se cu succes `cmd.exe`. În orice caz, capabilităţile interne ale interpretoarelor de comenzi sunt limitate, drept pentru care o mare parte din funcţionalitatea pe care acestea o oferă se bazează pe utilitarele pe care acestea le apelează. La acest capitol, *Windows PowerShell* funcţioneză oarecum diferit, el tratând comenzile ca pe nişte obiecte în contextul platformei .NET, ce sunt trecute prin acelaşi interpretor, ceea ce imprimă un comportament unitar precum şi o formă consistentă a comenzilor. *PowerShell* la versiunea 1.0 suportă aproximativ 130 de comenzi, toate încorporate în interpretor.

0.3.2.2 Instalarea *Windows PowerShell*

Pe un sistem *Windows Server 2008* **nu** este necesară descărcarea de pe Internet a pachetului de instalare. *PowerShell* este disponibil ca *feature* al sistemului şi nu e necesară decât activarea sa (vezi secţiunea următoare).

Material cu drept de autor

În afară de *Windows Server 2008*, *PowerShell* mai poate fi instalat şi poate rula pe următoarele platforme:

- Windows XP *Service Pack* 2
- Windows 2003 *Service Pack* 1
- Windows *Vista*
- Orice versiune mai recentă a acestora

Pentru toate sistemele de operare de mai sus este necesară prezenţa **Microsoft .NET Framework** cel puţin la versiunea 2.0.

La momentul scrierii acestei cărţi, versiunea curentă de *PowerShell* ce poate fi descărcată de pe site-ul Microsoft este *Windows PowerShell* 2.0 CTP2[1] (*Community Technology Preview*).

Versiunea de *PowerShell* inclusă în *Windows Server 2008* este 1.0. Înaintea instalării unei noi versiuni, este necesară dezinstalarea celei vechi folosind *Add/Remove Programs* sau *Programs and Features* din *Control Panel*.

Pe versiunile de Windows pe 32 de biţi, *PowerShell* se instalează implicit în directorul `%SystemRoot%\System32\WindowsPowerShell\v1.0`.

Pe versiunile de Windows pe 64 de biţi, varianta *PowerShell* pe 32 de biţi se instalează în directorul `%SystemRoot%\SystemWow64\WindowsPowerShell\v1.0` iar varianta pe 64 de biţi, ca şi în cazul anterior, la `%SystemRoot%\System32\WindowsPowerShell\v1.0`.

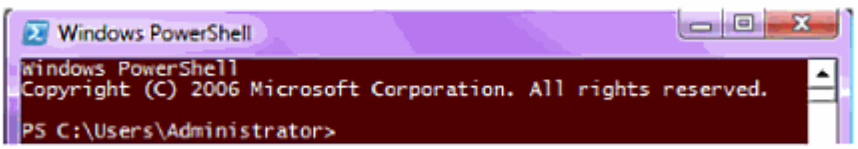
0.3.2.3 Lansarea în execuţie a *Windows PowerShell*

Pentru a minimiza riscurile de securitate, *Windows Server 2008* instalează implicit doar un set minimal de componente. De aceea, pentru a porni *PowerShell* pe *Windows Server 2008*, acesta trebuie instalat mai întâi ca *feature* al sistemului. Calea pentru activarea *PowerShell* ca *feature* este:

1. Se lansează *Server Manager*
2. În lista de *features* se selectează *Add feature*
3. Se bifează *Windows PowerShell* şi se confirmă instalarea

Odată instalat, sunt disponibile mai multe metode de a-l porni:

- Din fereastra de "`Run`", scriind *PowerShell* şi apăsând **Enter**
- Din interpretorul `cmd.exe`, scriind *PowerShell* şi apăsând **Enter**
- Din meniul `Start > All Programs > Windows PowerShell 1.0 > Windows PowerShell`



0-3: *Prompt*-ul *PowerShell*, după lansare

Deoarece *PowerShell* rulează într-o sesiune de consolă, poate fi lansat şi controlat şi de la distanţă, prin *Telnet* sau SSH. Pentru revenirea la promptul de comandă se tastează **exit**.

După lansare, se observă că interfaţa *PowerShell* este extrem de similară cu cea din `cmd.exe`, cu excepţia faptului că bara de titlu conţine acum textul *Windows PowerShell*, *prompt*-ul afişează caracterele *PS* la începutul liniei, iar fundalul ferestrei este albastru (cu excepţia cazului în care a fost lansat direct din `cmd.exe`).

[1] http://www.microsoft.com/technet/scriptcenter/topics/winpsh/pshell2.mspx

Material cu drept de autor

0.3.2.4 Configurări

Interfața *PowerShell*-ului poate fi configurată printr-un set relativ minimal de opțiuni disponibile prin clic-dreapta pe bara de titlu, la opțiunea *Properties*.

Printre parametrii ce pot fi configurați se numără: mărimea *buffer*-ului, tipul de font folosit și mărimea sa, mărimea ferestrei și a zonei ce poate fi derulată, precum și schema de culori folosită.

0.3.2.5 Comenzile *PowerShell*

PowerShell permite administratorilor să controleze și să automatizeze administrarea sistemului de operare, dar și a aplicațiilor ce rulează pe el. Comenzile sale sunt numite *„cmdlet(s)"* (pronunțat *„command-let(s)"* – în carte vor fi numite simplu „comenzi" de aici încolo) și permit accesul atât la sistemul de fișiere și la diverse resurse ale sistemului, precum și la zone ca *registry*, certificate digitale, etc.

„Limbajul" comenzilor *PowerShell* este unul orientat pe obiecte, înrudit cu limbajele de nivel înalt ce folosesc platforma .NET, cum ar fi C#, comun tuturor comenzilor, conceput pentru a simplifica sarcinile complexe, fără a adăuga un surplus inacceptabil de complexitate celor simple.

Pentru a lista toate opțiunile cu privire la parametrii disponibili la pornirea *PowerShell*, din `cmd.exe`, se folosește comanda:

```
PowerShell -?
```

Pentru a accesa ajutorul inclus în *PowerShell*, după pornirea acestuia se poate introduce comanda:

```
Get-Help
```

Majoritatea comenzilor *PowerShell* sunt simple, dar pot fi folosite în diverse combinații. Ele sunt usor identificabile după formatul numelor: un verb și un substantiv separate printr-o liniuță (-), de exemplu: `Get-Process`, `Start-Service`, `Get-Help`. Din formatul comenzilor se desprind câteva categorii:

- comenzile de tip „`get`" care returnează date;
- comenzile de tip „`set`" care introduc sau modifică date;
- comenzile de tip „`out`" care direcționează ieșirea spre o destinație specificată;
- comenzile de tip „`format`" care schimbă formatul datelor returnate ca rezultat.

Pentru a afișa o listă completă a comenzilor suportate de *PowerShell* se poate folosi comanda[1]:

```
Get-Command
```

Fiecare comandă include o secțiune de ajutor care descrie comportamentul, sintaxa și parametrii comenzii, alături de posibile exemple de utilizare, ca în cazul paginilor `man` din mediul UNIX. Pentru a accesa informațiile de ajutor ale unei comenzi se folosește sintaxa: `Get-Help <nume_comanda> -Detailed`, ca în exemplul următor:

```
PS C:\Users\Administrator> get-help get-help -detailed

NAME
    Get-Help

SYNOPSIS
    Displays information about Windows PowerShell cmdlets and concepts.
SYNTAX
```

[1] Se observă că în rezultatul lui `Get-Command` sunt returnate doar acele comenzi specifice și suportate nativ de către *PowerShell*, dar nu sunt listate toate comenzile suportate de acesta prin alias-uri sau din motive de compatibilitate cu vechile comenzi DOS sau UNIX. Mai multe detalii în secțiunea 0.3.2.10.

```
[...]
DETAILED DESCRIPTION
[...]
PARAMETERS
[...]
```

0-4: Ajutor pentru comanda Get-Help

În exemplul următor, comanda afişează o listă cu toate fişierele de ajutor din *Windows PowerShell*:

```
C:\PS>get help *
```

În legătură cu lansarea programelor externe din *PowerShell*, trebuie avut în vedere faptul că acesta împrumută aproape complet funcţionalitatea oferită de `cmd.exe`, spre exemplu posibilitatea lansării de aplicaţii atât în linie de comandă cât şi a celor grafice (*calculator*, *notepad*, etc) *PowerShell* identificând executabilele de lansat în directoarele specificate de variabila de mediu `%PATH%`, la fel ca şi `cmd.exe`. De asemenea, redirectările funcţionează identic. Afişarea variabilei de mediu `%PATH%` se poate face cu *PowerShell* prin comanda

```
Get-Content env:path.
```

Pentru `cmd.exe`, comanda similară este:

```
echo %path%.
```

De asemenea, în *PowerShell*, se poate folosi completarea automată a comenzilor pentru a cicla între variabilele de mediu disponibile. Pentru aceasta, se apasa tasta **Tab** după cele două pucte de după „`env`". În `cmd.exe`, pentru a returna o listă a variabilelor de mediu definite, împreună cu valorile lor, se foloseşte comanda „`set`".

0.3.2.6 Comenzi de ajutor în *PowerShell*

Obţinerea de ajutor pentru comenzile din *PowerShell* se face prin comanda:

```
Get-Help
```

Introducerea comenzii fără parametri are ca efect afişarea de informaţii legate de modul de utilizare al comenzii `Get-Help`.

Pentru a obţine ajutor despre o anumită comandă se introduce comanda respectivă ca parametru:

```
Get-Help Get-Process
```

Pentru a se afişa informaţii detaliate despre o comandă, se foloseşte parametrul `detailed`:

```
Get-Help Get-Process -Detailed
```

Dacă se doreşte afişarea tuturor informaţiilor de ajutor pentru o anumită comandă, la un nivel de detaliu puţin mai tehnic, se foloseşte parametrul `Full`:

```
Get-Help Get-Process -Full
```

Este posibilă afişarea doar a unei categorii de informaţii din pagina de ajutor. În exemplul următor, se afişează doar exemplele de utilizare ale comenzii `Get-Process`:

```
Get-Help Get-Process -Examples
```

De asemenea, pentru afişarea tuturor parametrilor unei comenzi, se foloseşte parametrul `Parameter` urmat de `*`. Intuitiv, pentru afişarea informaţiilor despre un singur parametru, acesta se specifică explicit:

```
Get-Help Get-Process -Parameter *
Get-Help Get-Process -Parameter id
```

Material cu drept de autor

Pentru a crea o consistenţă faţă de comenzile UNIX, comanda `Get-Help` poate fi înlocuită de comenzile:

```
man <nume_comanda>
```

sau

```
info <nume_comanda>
```

ce reprezintă funcţii interne ale *PowerShell*-ului care apelează `Get-Help` şi pot fi privite ca pe nişte alias-uri.

Caracterul * poate fi folosit în cadrul comenzii `Get-Help` pentru a afişa rezultatele mai multor comenzi. Spre exemplu, următoarea comandă afişează toate secţiunile de ajutor ce încep cu „*about*", împreună cu o scurtă descriere a lor:

```
Get-Help about*
```

Pentru mai multe detalii legate de modul în care pot fi substituite şirurile de caractere în *PowerShell* (şi în Windows, în general), se poate obţine ajutor direct din interiorul *PowerShell*-ului introducând comanda (totodată un exemplu pentru comanda anterioară):

```
Get-Help about_Wildcards
```

Wildcard-ul poate fi folosit şi în cadrul parametrilor. În exemplul următor se listează doar procesele care încep cu „w" din lista returnată de `Get-Process`:

```
PS C:\Users\Administrator> Get-Process w*

Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
    492      17    56868       9584   184    10.51   9308 winamp
    108       5     1580       2320    51     1.54    556 wininit
    152       4     2604       4176    53     1.70    604 winlogon
    638      34    49580      26888   342   785.89  10848 WINWORD
    150       5     2120       3164    48    10.58   1416 wlanext
    320      10    24192       5000   155    78.92   3508 WLTRAY
     40       2      680        996    21     0.02   1408 WLTRYSVC
```

0.3.2.7 Comenzi şi obiecte

Comenzile şi rezultatele din *PowerShell* sunt, de fapt, **obiecte .NET**. Conform conceptelor de bază ale programării orientate pe obiecte, un obiect reprezintă o instanţă, o entitate ce deţine anumite caracteristici descrise de proprietăţi şi poate executa anumite acţiuni prin intermediul metodelor.

Spre exemplu, o comanda ce returnează un serviciu in *PowerShell* de fapt returnează un obiect ce reprezintă serviciul. Informaţiile despre serviciu sunt proprietăţi ale obiectului returnat. Pornirea unui serviciu se traduce în setarea proprietăţii „*Status*" pe valoarea „*started*" folosind o metodă a obiectului serviciu.

Din faptul că fundamentul comenzilor *PowerShell* îl reprezintă obiectele, se deduce faptul că anumite utilitare de procesare bazate pe text şi aplicate pe rezultatele comenzilor *PowerShell* ar putea să nu aibă rezultatul aşteptat. De fapt, în cele mai multe dintre cazuri, folosirea utilitarelor de procesare text nici nu este necesară, din moment ce datele specifice pot fi extrase din rezultatele comenzilor folosind metodele standard de manipulare a obiectelor. Acest lucru este posibil deoarece rezultatele comenzilor încapsulează o mulţime de alte informaţii decât cele vizibile ca şiruri de caractere pe ecran.

Spre exemplu, o modalitate de a afişa informaţii despre toate interfeţele de reţea din sistem, fizice sau virtuale, o reprezintă următoarea comandă:

```
get-wmiobject Win32_NetworkAdapterConfiguration
```

Fără a i se aplica vreun filtru, comanda de mai sus afişează sub forma unei liste câţiva parametri ai interfeţelor, ca prezenţa unei adrese IP, a unui server DNS, dacă interfaţa este

Material cu drept de autor

configurată automat prin DHCP sau nu, numele său, etc, concatenând aceste informații pentru toate interfețele. La nivel de text, dacă s-ar dori doar afișarea numelui interfeței și a adresei IP configurate pe ea, s-ar folosi aceeași comandă, filtrând rezultatul său și păstrând același mod de afișare. În *PowerShell*, însă, rezultatul este un obiect din care se pot extrage doar proprietățile dorite, ca în exemplul următor:

```
      get-wmiobject  Win32_NetworkAdapterConfiguration  |  Select-Object  -property  IPAddress,
Description
```

Comanda de mai sus selectează doar proprietățile denumite `IPAddress` și `Description` pentru fiecare interfață și le afișează sub forma unui tabel.

O importantă comandă pentru afișarea tuturor proprietăților și metodelor unui obiect o reprezintă `Get-Member`. Spre exemplu, pentru obiectul returnat de comanda de mai sus, proprietățile și metodele sale pot fi afișate trimițându-l printr-un *pipe* comenzii `Get-Member`:

```
      PS C:\Users\Administrator> get-wmiobject Win32_NetworkAdapterConfiguration | get-member


         TypeName:
System.Management.ManagementObject#root\cimv2\Win32_NetworkAdapterConfiguration

      Name                          MemberType   Definition
      ----                          ----------   ----------
      DisableIPSec                Method       System.Management...
      EnableDHCP                    Method       System.Management...
      EnableIPSec                   Method       System.Management...
      EnableStatic                  Method       System.Management...
      ReleaseDHCPLease              Method       System.Management...
      DefaultTOS                    Property     System.Byte DefaultTOS {get;set;}
      DefaultTTL                    Property     System.Byte DefaultTTL {get;set;}
      Description                   Property     System.String Description {get;set;}
      DHCPEnabled                     Property     System.Boolean DHCPEnabled {get;set;}
      DHCPLeaseExpires                Property     System.String DHCPLeaseExpires {get;set;}
      DHCPLeaseObtained             Property     System.String DHCPLeaseObtained {get;set;}
      DHCPServer                    Property     System.String DHCPServer {get;set;}
      DNSDomain                     Property     System.String DNSDomain {get;set;}
      [...]
```

0.3.2.8 Înlănțuirea comenzilor (*pipes*)

PowerShell permite înlănțuirea mai multor comenzi prin folosirea operatorului *pipe* (|) cu același efect ca și în UNIX: redirectarea ieșirii unei comenzi spre intrarea alteia. Datorită modului în care anumite comenzi acceptă date sau din considerente ale formatului dorit la ieșire, deseori sunt necesare prelucrari la nivel de șiruri de caractere între două comenzi ce prelucrează date sau înainte de afișarea rezultatului final.

În mod practic, ieșirile și intrările comenzilor sunt tratate ca obiecte, astfel că o comandă ce primește ca intrare rezultatul alteia poate acționa direct asupra proprietăților și metodelor sale fără conversii suplimentare. De asemenea, utilizatorul poate adresa proprietăți și metode direct după nume, nefiind necesară filtrarea rezultatului pentru selectarea datelor dorite, ca în cazul UNIX-ului și a utilitarelor `sed`, `awk`, `grep`, `cut`, șamd.

În exemplul următor, rezultatul unei comenzi `ipconfig` este trimis comenzii `findstr` care primește ca parametru șirul de caractere de căutat. Rezultatul este identic cu al utilitarului `grep` din UNIX, folosit în condiții similare:

```
      PS C:\Users\Administrator> ipconfig | findstr "Address"
         Link-local IPv6 Address . . . . . : fe80::a928:ea78:f920:ac15%10
         IPv4 Address. . . . . . . . . . . : 141.85.37.193
         Link-local IPv6 Address . . . . . : fe80::9dd5:cc72:ab31:8927%12
         IPv4 Address. . . . . . . . . . . : 192.168.13.1
         Link-local IPv6 Address . . . . . : fe80::525:fc37:994d:636f%14
         IPv4 Address. . . . . . . . . . . : 192.168.216.1
         IPv6 Address. . . . . . . . . . . : 2002:8d55:25c1::8d55:25c1
```

0-5: Folosirea operatorului "|" pentru filtrarea rezultatului unei comenzi

Material cu drept de autor

0.3.2.9 Folosirea parametrilor

Parametrii in *PowerShell* sunt identificaţi prin caracterul „-". Caracterele („/" sau „\") ce marcau parametrii în `cmd.exe` nu mai sunt folosite. La tastarea unui parametru se poate introduce numele complet, dar este acceptat şi dacă se introduce un număr suficient de caractere pentru a distinge parametrul de altele cu nume asemănătoare. În exemplul următor, cele două comenzi sunt identice, şirul *„det"* fiind suficient pentru a-l distinge de parametrul *Debug*:

```
Get-Help Get-Date -Detailed
Get-Help Get-Date -det
```

În unele cazuri, introducerea numelor parametrilor este opţională, fiind suficientă introducerea valorii lor. Spre exemplul, forma completă a comenzii `Get-Help` presupune declararea numelui comenzii precedat de parametrul *Name*, ca în exemplul de mai jos. Totuşi, parametrii pot fi folosiţi fără a li se specifica numele dacă se respectă cu strictețe ordinea lor din secţiunea de sintaxă a paginii de ajutor:

```
Get-Help -Name Get-Host
```

0.3.2.10 Interacţiunea cu *Windows PowerShell*

În general, comenzile *PowerShell* nu sunt *case-sensitive*. Totuşi, pentru situaţiile în care se specifică valori literale, ca în cazul în care se dă ca parametru un şir de caractere de căutat, capitalizarea corectă poate avea importanţă.

Pentru a facilita interacţiunea, *PowerShell* oferă opţiunea de completare automată contextuală. Aceasta poate fi folosită prin apăsarea tastei **Tab** atât pentru numele comenzilor cât şi pentru parametrii lor, din momentul în care s-a introdus cel puţin un caracter din termenul de completat. Completarea automată realizează automat capitalizarea corectă a comenzilor.

Odată ce *PowerShell* a fost lansat, poate fi folosit în aceeaşi manieră ca şi interpretorul `cmd.exe`. De exemplu, comanda `dir` listează conţinutul unui director, `cd` schimbă directorul curent şi pot fi folosite comenzi precum `copy`, `move`, `del`, şamd. Aşa cum s-a mai menţionat şi în secţiunea 0.3.2.6: *Comenzi de ajutor în PowerShell*, listarea comenzilor încorporate în *PowerShell* se poate face prin comanda `Get-Command`. Totuşi, *PowerShell* acceptă o serie mult extinsă de comenzi prin faptul că implementează o multitudine de alias-uri de comenzi care înlocuiesc una sau mai multe comenzi înlănţuite, din considerente de uzabilitate, eficienţă în compunerea comenzilor sau pentru a păstra similarităţi cu DOS şi UNIX. Lista completă a acestor comenzi poate fi obţinută prin comanda `help` sau, indirect, prin comanda `Get-Help *` care are drept efect listarea tuturor fişierelor de ajutor pentru comenzile corespunzătoare[1]. Aceasta asigură, intuitiv, faptul că toate comenzile, atât încorporate cât şi aliasurile sunt documentate.

Câteva dintre comenzile UNIX ce sunt implementate în *PowerShell* ca aliasuri includ: `diff`, `ps`, `ls`, `pwd`, `cat`, etc.

Spre exemplu, aliasul `ps` are acelaşi efect ca şi comanda `Get-Process`, ca în exemplul următor:

```
PS C:\> ps

Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
    243      12   247672     257788   368   435.57   7344 AcroRd32
    127      14     1796       2004    39     0.22   2544 alg
```

[1] Un fapt interesant de remarcat este că la utilizarea comenzii `help`, rezultatul este afişat în stilul utilitarului `more`, din UNIX: tasta Enter pentru linia următoare sau Space pentru ecranul următor.

Material cu drept de autor

```
     65       3     2172        796    38     0.12   2008 AppleMobileDeviceService
    109       3     1216        968    36     0.14    988 Ati2evxx
    133       4     1956       3460    51    79.33   1952 Ati2evxx
    154       6    24324      16160    71           1572 audiodg
     47       2     1120        472    49     0.11   3452 brs
    187      10     8288       7732   102    33.62   1828 BTStackServer
[...]

PS C:\> Get-Proces

Handles  NPM(K)    PM(K)      WS(K) VM(M)   CPU(s)     Id ProcessName
-------  ------    -----      ----- -----   ------     -- -----------
    243      12   247672     257788   368   435.57   7344 AcroRd32
    127      14     1796       2004    39     0.22   2544 alg
     65       3     2172        796    38     0.12   2008 AppleMobileDeviceService
    109       3     1216        968    36     0.14    988 Ati2evxx
    133       4     1956       3460    51    79.33   1952 Ati2evxx
    154       6    24324      16160    71           1572 audiodg
     47       2     1120        472    49     0.11   3452 brs
    187      10     8288       7732   102    33.62   1828 BTStackServer
[...]
```

0-6: *ps* (alias) şi *Get-Process* (comandă)

Pentru a demonstra că rezultatul aliasului este, de asemenea, de tip obiect, folosind informaţiile din acesta, se pot ordona procesele după număr şi exporta lista într-un fişier în format HTML[1]:

```
PS C:\> ps | Sort-Object id | ConvertTo-Html > procese.html
PS C:\> ls procese.html


    Directory: Microsoft.PowerShell.Core\FileSystem::C:\


Mode                 LastWriteTime     Length Name
----                 -------------     ------ ----
-a---          8/1/2008   4:04 PM     198444 procese.html
```

PowerShell oferă comenzi şi pentru interacţiunea cu WMI (*Windows Management Instrumentation*). Comanda de bază folosită pentru a obţine obiecte prin WMI este `Get-WmiObject`, cu *alias*-ul *„gwmi"*). Spre exemplu, obţinerea de informaţii despre BIOS-ul sistemului se poate face facil prin comenzile `Get-WmiObject -Class Win32_BIOS` sau `Get-WmiObject -Class Win32_ComputerSystem`.

```
PS C:\> Get-WmiObject -Class Win32_BIOS


SMBIOSBIOSVersion : A14
Manufacturer      : Dell Inc.
Name              : Phoenix ROM BIOS PLUS Version 1.10 A14
SerialNumber      : 3YNOW2J
Version           : DELL   - 27d70402

PS C:\> Get-WmiObject -Class Win32_ComputerSystem


Domain              : WORKGROUP
Manufacturer        : Dell Inc.
Model               : MM061
Name                : WIN-8J4PG74KHA9
PrimaryOwnerName    : Windows User
TotalPhysicalMemory : 2145083392
```

De asemenea, WMI poate fi folosit şi pentru interacţiunea cu procese sau servicii. Listarea lor se face prin comenzile: `Get-WmiObject win32_process`, respectiv `Get-WmiObject win_32_service`. Atenţie, aceste comenzi au rezultate extrem de voluminoase, din moment ce afişează toate informaţiile ce pot fi extrase din obiectele de tip proces, respectiv serviciu. O soluţie pentru reducerea cantităţii de informaţii afişate o reprezintă fie aplicarea unui filtru, fie

[1] Se observă apoi, la vizualizarea paginii HTML obţinute, că în aceasta au fost incluse sub forma capetelor de tabel toate datele despre procese ce sunt returnate de comanda `ps` / `Get-Process`, mai exact, 62 de parametrii specifici fiecărui proces, în comparaţie cu cei 8 care sunt afişaţi implicit în consolă, ceea ce confirmă încă o dată natura de obiect a rezultatelor comenzilor *PowerShell*.

Material cu drept de autor

specificarea doar a anumitor câmpuri ce vor fi afşate, împreună cu valorile lor. În următorul exemplu sunt interogate doar anumite proprietăți ale obiectului ce încapsulează informații despre sistemul de operare curent, obținut prin WMI:

```
PS C:\> Get-WMIObject -class Win32_OperatingSystem -property csname, Caption, Version,
TotalVirtualMemorySize, FreeVirtualMemory


__GENUS                : 2
__CLASS                : Win32_OperatingSystem
[...]
__RELPATH              : Win32_OperatingSystem=@
[...]
Caption                : Microsoft® Windows Server® 2008 Standard
CSName                 : WIN-8J4PG74KHA9
FreeVirtualMemory      : 2602080
TotalVirtualMemorySize : 4471252
Version                : 6.0.6001
```

0-7: Selectarea explicită a anumitor câmpuri ale unui obiect returnat

În mod evident, interacțiunea cu procese şi servicii prin *PowerShell* şi WMI nu se reduce doar la afişarea de informații. În exemplul următor se demonstrează modul în care un serviciu poate fi pornit, împreună cu verificarea stării acestuia înainte şi după pornire:

1. Se caută un serviciu cu numele asemănător cu „*ipod*":

```
PS C:\> Get-Service ipod*

Status   Name               DisplayName
------   ----               -----------
Running  iPod Service       iPod Service
```

2. După identificarea sa, i se interoghează starea curentă:

```
PS C:\> Get-WmiObject win32_service -filter "name='ipod service'"


ExitCode  : 1077
Name      : iPod Service
ProcessId : 0
StartMode : Manual
State     : Stopped
Status    : OK
```

3. Se observă că serviciul este oprit. Se foloseşte metoda `StartService()` din obiectul de tip serviciu returnat prin WMI pentru a-l porni:

```
PS C:\> (Get-WmiObject win32_service -filter "name='ipod service'").StartService()


__GENUS          : 2
__CLASS          : __PARAMETERS
__SUPERCLASS     :
__DYNASTY        : __PARAMETERS
__RELPATH        :
__PROPERTY_COUNT : 1
__DERIVATION     : {}
__SERVER         :
__NAMESPACE      :
__PATH           :
ReturnValue      :
```

4. Se verifică din nou starea serviciului şi se observă că acesta rulează fără probleme:

```
PS C:\> Get-WmiObject win32_service -filter "name='ipod service'"


ExitCode  : 0
Name      : iPod Service
ProcessId : 904
StartMode : Manual
State     : Running
Status    : OK
```

Următorul exemplu se încadrează într-o categorie de utilizare mai „exotică" a comenzilor ce se folosesc de WMI pentru a obține informații. Folosind parametrul `query`, se poate

Material cu drept de autor

specifica sursa din care se extrag datele, câmpurile de interes precum şi ce filtre se vor aplica acestora, într-un printr-o sintaxă SQL:

```
PS C:\> get-wmiobject -query "select * from win32_service where name='dhcp'"


ExitCode  : 0
Name      : Dhcp
ProcessId : 1008
StartMode : Auto
State     : Running
Status    : OK
```

0-8: Obţinerea stării unui serviciu folosind *PowerShell* şi WMI, printr-o comandă SQL

Există, bineînţeles, o multitudine de alte moduri în care *PowerShell* poate fi folosit, scurtături şi excepţii în utilizare, dar detalierea acestora ar putea crea subiectul unei cărţi de sine stătătoare, aşadar acestea vor fi excluse dintre subiectele tratate în paginile de faţă. Totuşi, pentru a putea aplica *PowerShell* în utilizarea diverselor servicii ce vor fi detaliate în capitolele următoare, e necesară aprofundarea la un nivel mediu a tehnicilor de *scripting* în *PowerShell*.

0.3.2.11 *Scripting* în *PowerShell*

În mod normal, e de aşteptat ca orice interpretor de comenzi să suporte, nativ sau nu, posibilitatea de a rula scripturi. Posibilitatea de *scripting* a existat şi în trecut, prin intermediul lui `cmd.exe`, dar capabilităţile lui *PowerShell* se ridică mult deasupra acestora. Totodată, prin *scripting* se observă mult mai clar interdependenţa dintre *PowerShell* şi platforma .NET, folosindu-se convenţii de denumire şi o sintaxă mult asemănătoarelor limbajelor bazate pe .NET, precum C#.

0.3.2.11.1 Concepte de securitate în scripting

Posibilitatea de a automatiza şi drepturile pe care le au în general scripturile de a interacţiona cu sistemul au creat dintotdeauna numeroase dificultăţi şi au dat naştere unui mediu propice pentru apariţia viruşilor, viermilor şi a aplicaţiilor *spyware*. Câteva considerente de securitate au fost implementate implicit în *PowerShell* tocmai pentru a adresa limitările impuse de aceste riscuri:

- Scripturile *PowerShell* nu sunt inerent executabile. În Windows, aceasta înseamnă că nu există o asociere implicită între *PowerShell* şi scripturile sale care au extensia `.ps1`. Un dublu-clic pe acest tip de fişier nu va cauza execuţia sa, ci va fi deschis cu *Notepad* pentru a i se vizualiza conţinutul.
- Singurele scripturi acceptate spre a fi rulate sunt cele semnate şi autorizate prin colecţia de certificate a sistemului.
- Pentru scripturile asupra cărora există dreptul de execuţie, aceasta poate fi iniţiată doar dacă se specifică întreaga cale a scriptului, pentru a evita instalarea de scripturi cu nume de comenzi uzuale în anumite directoare (un echivalent al *rootkit*[1]-urilor de pe Linux).

Evident, aceste măsuri nu sunt suficiente dar oferă un grad sporit de evitare a procedurilor ce ar putea cauza daune sistemului. Drepturile care stabilesc regulile după care scripturile sunt sau nu executate sunt dictate de politica de execuţie activă în sistem. Pentru a vedea care este aceasta, se foloseşte comanda următoare:

```
PS C:\> Get-ExecutionPolicy
Restricted
```

[1] http://en.wikipedia.org/wiki/Rootkit şi http://www.rootkit.com

Material cu drept de autor

În exemplul de mai sus, politica activă este cea de *Restricted*, care este şi cea implicită. Valorile posibile pe care politica de execuţie a sistemului le poate avea sunt:

- ***Restricted***: implicit, nu acceptă execuţia niciunui script
- ***AllSigned***: sunt acceptate spre execuţie doar scripturile semnate
- ***RemoteSigned***: este permisă execuţia scripturilor locale; restul trebuie semnate
- ***Unrestricted***: toate scripturile pot fi executate

Politica minimă necesară pentru a putea rula local scripturi nesemnate este *RemoteSigned*. Modificarea politicii curente de execuţie în *RemoteSigned* se face cu următoarea comandă:

```
PS C:\> Get-ExecutionPolicy
Restricted
PS C:\> Set-ExecutionPolicy RemoteSigned
PS C:\> Get-ExecutionPolicy
RemoteSigned
```

0.3.2.11.2 Exemplu: crearea primului script în PowerShell

Pentru a crea un script *PowerShell*, se creează un fişier text cu *Notepad* sau orice editor de text şi i se setează extensia `ps1`. Setarea unei alte extensii va avea ca efect, la rularea sa, tentativa de a fi tratat de către Windows cu o aplicaţie implicită ce acceptă extensia respectivă şi nu va fi executat de către *PowerShell*.

1. Se creează un fişier text cu următorul conţinut:

```
$a = "Hello"
$b = "World!"
write-host $a
echo $b
```

2. Se salvează cu extensia `ps1`. Pentru exemplu se va considera că se salvează ca `C:\test.ps1`.
3. În *PowerShell*, se introduce următoarea comandă şi se primeşte rezultatul[1]:

```
PS C:\> PowerShell c:\test.ps1
Hello
World!
```

0.3.2.11.3 Variabile

Conceptul de variabilă se întâlneşte în aproape orice limbaj de programare sau de *scripting*. În esenţă, reprezintă o zonă de memorie adresabilă printr-un nume dat de programator care poate să stocheze date ce vor fi folosite la un moment ulterior. În *PowerShell* pot fi definite variabile, dar numele acestora trebuie să înceapa cu semnul dolar ($). Numele variabilelor pot conţine caractere alfanumerice (litere şi cifre) şi alte câteva simboluri. Se pot introduce chiar şi spaţii în numele variabilelor, cu condiţia ca numele să fie cuprinse între paranteze acolade {}.

Următoarele exemple reprezintă initializări valide de variabile:

```
$name = "Nicoleta"
$pi = 3.14
${variabila cu spatii}   "atentie la paranteze!"
```

Atribuirea variabilelor cu o valoare se putea face şi în trecut, prin intermediul lui `cmd.exe` şi al comenzii `SET`, cu diferenţa că acesta din urmă nu suporta denumirea variabilelor folosind spaţii.

După cum se observă, nu s-a specificat nicăieri tipul datelor ce sunt stocate în variabile. Acest lucru poate crea uneori probleme în scripturi complexe, de aceea *PowerShell* oferă şi posibilitatea de a defini variabile cu tip. Cu alte cuvinte, *PowerShell* poate fi informat despre

[1] `echo` este un alias pentru comanda `Write-Output`.

Material cu drept de autor

tipul datelor ce vor fi reținute în variabila respectivă, ceea ce reprezintă și o practică recomandată. Fie, spre exemplu, codul următor:

```
$a = 5
write-host $a + 3
```

În secvența anterioară s-a atribuit valoarea 5 variabilei `$a`, după care s-a afișat în consolă rezultatul expresiei `$a + 3`. După cum e de așteptat, se va afișa 8.

Dar dacă se ia în considerare exemplul următor:

```
$a = 7
$s = "un sir de caractere"
…cod nesemnificativ…
$a = "Nicoleta"
…în continuare, cod nesemnificativ…
write-host ($a + 7)
```

Codul de mai sus definește variabila `$a` cu valoarea 7 și variabila `$s` cu șirul de caractere „Nicoleta". Presupunând că atribuirea s-a făcut în mod eronat între variabila `$a` și șirul „Nicoleta", rezultatul afișat în urma expresiei de pe ultima linie va consta în concatenarea celor două elemente ca șiruri de caractere, rezultând „Nicoleta7". Această problemă poate fi ușor evitată prin stabilirea tipurilor de date stocate în fiecare variabilă, împiedicând operația de atribuire dintre o variabilă definită ca întreg (*integer*) și un șir de caractere (*string*), ca în exemplul următor, rescris:

```
[int]$a = 7
[string]$s = "un sir de caractere"
…cod nesemnificativ…
$a = "Nicoleta"
…în continuare, cod nesemnificativ…
write-host ($a + 7)
```

Stabilirea tipurilor variabilelor se face prefixându-le numele cu un identificator ce descrie un tip de date, în exemplul de mai sus, `[int]` și `[string]`. Rularea exemplului de mai sus va cauza o eroare, ce va anunța faptul că membrul drept al atribuirii `$a = "Nicoleta"` nu se află în formatul corespunzător.

Cele mai comune tipuri de variabile utilizabile în *PowerShell* sunt următoarele[1]:

- `[boolean]` Adevărat sau fals.
- `[int]` Numere întregi pe 32 de biți.
- `[char]` Un singur caracter.
- `[string]` Șir de caractere.
- `[single]` Număr zecimal, simplă precizie.
- `[double]` Număr zecimal, dublă precizie.
- `[datetime]` Data, ora.
- `[adsi]` Obiect de tip *Active Directory* Service Interface[2] (ASDI)
- `[wmi]` Instanță sau colecție WMI
- `[wmiclass]` Clasă WMI

Vectorii în *PowerShell* se definesc extrem de ușor, ca în exemplul de cod următor:

```
$PS C:\> $vec = 4,8,15,16
PS C:\> Write-Host $vec
4 8 15 16
```

Pentru referirea la elementele unui vector, se folosește adresarea indexată, ținând cont că indecșii elementelor dintr-un vector încep de la 0 (zero), ca în majoritatea limbajelor de programare și scripting: `$vec[4]`, `$vec[0]`, etc.

[1] Există și variabila predefinită `$NULL`. Orice variabilă neinițializată (neutilizată) are conținutul egal cu `$NULL`.

[2] Mai multe despre ASDI: http://msdn.microsoft.com/en-us/library/aa772170.aspx

Material cu drept de autor

Adăugarea de noi elemente într-un vector se face tot prin intermediul operatorului „+”[1]:

```
PS C:\> $vec = $vec + 23, 48
PS C:\> Write-Host $vec
4 8 15 16 23 48
```

0.3.2.11.4 *Expresii condiționale*

Ca în orice limbaj de programare sau de *scripting*, flexibilitatea sa stă în posibilitatea de a decide ramurile ce vor fi executate în funcție de valorilor anumitor variabile. În fond, condiția este ceea ce stă la baza scripturilor şi le diferențiază de o simplă înlănțuire inalterabilă de comenzi care se execută întotdeauna în aceeaşi ordine.

Exemplul următor ar trebui să poată fi deja urmărit fără prea multe explicații suplimentare:

```
$a = 2
if ($a -eq 1)
{
        write-host "unu"
}
elseif ($a -eq 2)
{
        write-host "doi"
}
else
{
        write-host "diferit de unu sau doi"
}
```

Fragmentul de script de mai sus verifică valoarea atribuită variabilei `$a`. Dacă aceasta este 1 (numeric) va afişa „`unu`”, daca este 2 va afişa „`doi`” şi pentru orice altă valoarea (inclusiv valori non-numerice şi alte tipuri) va afişa „`diferit de unu sau doi`”[2].

După cum se observă, singurul element relativ nou introdus este operatorul de egalitate folosit în interiorul *if*-ului: `-eq`. Există o serie de alți operatori de comparație ce pot fi folosiți în *PowerShell*, fiecare dintre aceştia fiind descrişi printr-o cratimă (-) urmată de o abreviere de două litere a funcției numerice pe care acesta o îndeplineşte precum şi alte forme pentru diferite tipuri de date. Lista este următoarea:

- **`-eq`** Egalitate
- **`-ne`** Inegalitate
- **`-lt`** Mai mic
- **`-le`** Mai mic sau egal
- **`-gt`** Mai mare
- **`-ge`** Mai mare sau egal
- **`-contains`** Determină apartenența la un grup, retunează întotdeauna `[boolean]True` sau `[boolean]False`
- **`-notcontains`** Determină neapartenența la un grup, retunează întotdeauna `[boolean]$True` sau `[boolean]$False`
- **`-match`** Consistență la compararea prin expresii regulate
- **`-notmatch`** Inconsistență la compararea prin expresii regulate
- **`-like`** Consistență la compararea cu *wildcard*-uri
- **`-notlike`** Inconsistență la compararea cu wildcard-uri
- **`-band`** Şi binar
- **`-bor`** SAU binar
- **`-bnot`** negație binară
- **`-is`** Este de tipul (ex: `$x -is [int]`)
- **`-isnot`** Nu este de tipul (ex: `$x -isnot [single]`)

[1] În general, expresiile de tipul $x = $x + y pot fi înlocuite prin expresia $x += y, ca şi în limbajul C.

[2] Paranteze acolade sunt obligatorii în toate situațiile, chiar dacă încadrează o singură instrucțiune.

În plus, *PowerShell* implementează variantele *case sensitive* ale unor comenzi prezentate mai sus (`-clt`, `-cgt`, `-cle`, `-cge`, `-ceq`, `-cne`, `-clike`, `-cnotlike`, `-ccontains`, `-cnotcontains`, `-cmatch`, `-cnotmatch`), precum şi o serie de operatori speciali, pe lângă cei clasici, ca +, –, /, %, *, !:

- **`-replace`** Înlocuire (ex: `"abcde" -replace "b", "B"`)
- **`-ireplace`** Înlocuire *case-insensitive*
- **`-as`** Convertire la alt tip (ex: `123 -as [string]` tratează 123 ca şir de caractere)
- `..` Operator de interval (ex: `foreach ($i in 1..10) {$i}` afişează numerele de la 1 la 10)
- **`&`** Operator de apel (ex: `$a = "Get-ChildItem" &$a` va executa Get-ChildItem)
- **`-F`** Operator de formatare (ex: `foreach($p in Get-Process) {"{0,-15} has {1,6} handles" -F $p.processname,$p.Handlecount}` are ca efect afişarea liste de procese în formatul `<nume_proces> has <x> handles`)

În *PowerShell* se poate folosi şi o construcţie de tipul *switch-case*, cu o sintaxă mai simplă decât în C:

```
$culoare = "rosu"
switch ($culoare)
{
   rosu {write-host "culoarea rosie"; break}
   verde {write-host "culoarea verde"; break}
   albastru {write-host "culoarea albastra"; break}
   galben {write-host "culoarea galbena"; break}
}
```

0.3.2.11.5 Instrucţiuni de ciclare

În *PowerShell* există 4 moduri în care se pot defini ciclurile (repetiţiile): `for`, `foreach`, `while` şi `do..while`.

Ciclul *for* are exact aceeaşi sintaxă ca şi în limbajul C, spre exemplu:

```
for(<iniţializări>;<condiţie>;<repetare>)
{
   <instrucţiuni>
}
```

Următorul cod afişează numerele de la 1 la 100:

```
for($i=1;$i -lt 101;$i++)
{
   write-host $i
}
```

Ca şi în C, secţiunea de `<iniţializări>` se execută doar o dată, la prima iteraţie, secţiunea de `<condiţie>` este verificată după fiecare iteraţie, iar secţiunea de `<repetare>` este executată la finalul fiecărei iteraţii. Se observă posibilitatea utilizării operatorului ++.

Spre deosebire de instructiunea `for`, instrucţiunea `foreach` este concepută astfel încât primind o colecţie de obiecte, va executa o secţiune de cod pentru (*for*) fiecare (*each*) dintre elementele din colecţie (de unde şi numele). Sintaxa generală este următoarea:

```
foreach ($<element> in $<colecţie_de_elemente>)
{
   <instrucţiuni>
}
```

Următorul exemplu foloseşte instrucţiunea `foreach` pentru a afişa numele tuturor fişierelor din directorul `C:\Windows\System32`:

```
foreach ($file in Get-ChildItem C:\Windows\System32)
{
   write-host $file
}
```

Material cu drept de autor

Din nou similară cu C, sintaxa unei construcții de tip `While` este următoarea:

```
while(<condiție>)
{
   <instrucțiuni>
}
```

Sintaxa lui `Do..While` este, de asemenea, similară cu C, cu excepția semnului punct și virgulă de după `While`, care în *PowerShell* lipsește[1]:

```
do
{
   write-host $a
   $a++
} while ($a -lt 10)
```

De asemenea, în construcțiile de tip repetiție pot fi folosite și instrucțiunile `break` și `continue`, cu efectele binecunoscute.

0.3.2.12 Aplicații *PowerShell*

După enumerarea unei bune părți a capabilităților de *scripting* ale lui *PowerShell*, pot fi analizate o serie de exemple de utilizare mai complexe, folosind structurile studiate mai sus.

Fie următoarea secvență de script:

```
Get-Service | ForEach {if ($_.Status -eq "Running") {write-host $_.DisplayName}}
```

Secvența de mai sus returnează numele întreg (nu al executabilului, ci descrierea) al tuturor serviciilor ce rulează în sistem.

O serie de construcții noi prezentate aici necesită explicare. În primul rând, `Get-Service` returnează lista obiectelor serviciilor din sistem. Rezultatul obținut din acesta este trimis ca intrare prin operatorul *pipe* (|) ciclului `foreach`. Din moment ce `Get-Service` returnează o colecție de obiecte de tip serviciu, acest tip de rezultat este perfect pentru se itera prin elementele sale folosind `foreach`. Folosind instrucțiunea `if` se verifică dacă starea fiecărui serviciu este *„Running"* și, în caz afirmativ, i se afișează numele complet.

În interiorul instrucțiunii `if` se remarcă două elemente noi: notația `$_` și *„."* (punctul). Variabila `$_` reprezintă o variabilă de sistem definită automat. În interiorul unui *pipe*, ea reține elementul curent din „fluxul" trecut prin acel *pipe* (elementul curent din *pipeline*). În exemplul de față, la fiecare iterație a repetiției, `$_` va referenția pe rând fiecare obiect de tip serviciu din colecția returnată de `Get-Service`. Notația *„."* reprezintă referirea la un membru al unui obiect (ca în Java sau în structurile C). Fiecare obiect are un set de proprietăți; spre exemplu, un obiect de tip serviciu deține proprietăți ca `status`, `name` și `displayname`, printre multe altele. Accesarea unei astfel de proprietăți se face prin operatorul punct astfel: `.<nume_proprietate>`.

La fel ca și `$_`, există și alte variabile definite automat în sistem:

- `$_` Conține obiectul curent din *pipe*.
- `$?` Conține valoarea *True* dacă ultima operație s-a încheiat cu succes și *False* altfel.
- `$Home` Directorul utilizatorului curent, echivalentul concatenării variabilelor de mediu `%homedrive%` și `%homepath%`
- `$LASTEXITCODE` Codul de ieșire al ultimei execuții
- `$PsHome` Directorul de instalare a *PowerShell*
- `$Host` Conține informații despre mediul în care rulează consola curentă

Fie exemplul următor:

[1] În exemplul `do..while` nu este arătată valoarea cu care este inițializată variabila `$a`. Implicit, valoarea acesteia este `$NULL`. Există anumite optimizări pentru lucrul cu această valoare, ca spre exemplu `$a++`, în condițiile în care `$a` este `$NULL`, va returna valoarea 1, dar nu este recomandabil lucrul cu variabile neinițializate.

Material cu drept de autor

```
Get-ChildItem D:\Logs\* -include *.txt |
foreach {move-item $_ ($_ -replace(".txt",".bak"))}
```

Codul de mai sus obţine o listă a tuturor fişierelor cu extensia `.txt` din directorul `D:\Logs\` şi o trimite unui ciclu `foreach` care iterează pe fiecare element şi rulează comanda `move-item` cu doi parametri: primul ramâne numele fişierului primit din *pipe* iar al doilea reprezintă tot numele fişierului curent, dar cu extensia `.txt` substituită în `.bak`. Cu alte cuvinte, comanda redenumeşte toate fişierele text din `D:\Logs\` schimbându-le extensia `.txt` în `.bak`.

0.3.2.13 *PowerShell* şi Windows Registry

Modificarea configuraţiei din Windows Registry poate fi deseori o sarcină a unui administrator. Deseori, pentru efectuarea de modificări în Registry se foloseşte utilitarul `regedit.exe` sau, în cazul scripturilor (non-*PowerShell*), se poate folosi şi utilitarul în linie de comandă `reg.exe`. Totuşi interacţiunea cu Windows Registry se poate face facil şi prin *PowerShell* deoarece acesta tratează Registry-ul ca pe un sistem de fişiere (lucru intuitiv, ţinând cont ca are o organizare arborescentă).

Una dintre cele mai des accesate chei din Registry este cheia `Run`, care stochează căile spre executabilele ce sunt lansate automat la pornirea sistemului[1]. Pentru a naviga spre aceasta se execută comenzile:

```
PS C:\Users\Administrator> cd HKLM:
PS HKLM:\> cd SOFTWARE\Microsoft\Windows\CurrentVersion\Run
```

După cum se observă, Windows Registry este considerat ca fiind o unitate (un *drive*) separat, numit HKLM (*Hive Key Local Machine*). Registry-ul este împărţit în mai multe zone logice, denumite *hive*-uri, ce poartă denumirile din Windows API[2] şi prescurtări ca HKLM (...*Local Machine*), HKCU (...*Current User*), HKCR (...*Classes Root*), HKU (...*Users*), etc.

Cu toate că Registry-ul este tratat ca un arbore, tentativa de a afişa conţinutul cheii `Run` printr-un simplu `dir` va returna doar rezultatul următor:

```
PS HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run> dir


   Hive:
Microsoft.PowerShell.Core\Registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersio
n\Run

SKC  VC Name                           Property
---  -- ----                           --------
  3   1 OptionalComponents             {(default)}
```

Acest lucru se întâmplă deoarece doar cheile sunt considerate obiecte, valorile lor fiind tratate ca proprietăţi. Aşadar, pentru obţinerea valorii unei chei se poate folosi comanda `Get-ItemProperty`, eventual cu parametrul „.”, adică directorul curent, pentru a afişa toate valorile cheilor din locaţia curentă:

```
PS HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run> Get-ItemProperty .

[...]
PSChildName                 : Run
PSDrive                     : HKLM
PSProvider                  : Microsoft.PowerShell.Core\Registry
IntelliPoint                : "C:\Program Files\Microsoft IntelliPoint\ipoint.exe"
SigmatelSysTrayApp          : sttray.exe
Windows Defender            : C:\Program Files\Windows Defender\MSASCui.exe -hide
SynTPEnh                    : C:\Program Files\Synaptics\SynTP\SynTPEnh.exe
vmware-tray                 : C:\Program Files\VMware\VMware Workstation\vmware-tray.exe
```

[1] Este vorba despre cheia `Run` din `HKEY_LOCAL_MACHINE`. Excepţia o reprezintă cheia `Run` din `HKEY_CURRENT_USER` care specifică executabilele ce vor fi rulate doar când utilizatorul se autentifică în sistemul de operare.

[2] http://msdn.microsoft.com

Material cu drept de autor

```
VMware hqtray                : "C:\Program Files\VMware\VMware Workstation\hqtray.exe"
googletalk                   : C:\Program Files\Google\Google Talk\googletalk.exe /autostart
GrooveMonitor                : "C:\Program Files\MicrosoftOffice\Office12\GrooveMonitor.exe"
Broadcom Wireless Manager UI : C:\Windows\system32\WLTRAY.exe
SunJavaUpdateSched           : "C:\Program Files\Java\jre1.6.0_06\bin\jusched.exe"
```

0-9: Rezultatul afişării cheii *Run* folosind *PowerShell*

În exemplele anterioare s-a lucrat direct în interiorul *hive*-ului HKLM. Pentru a schimba locaţia curentă în unul dintre celelalte *hive*-uri, este necesară întâi conectarea la *PowerShell Registry Provider*, un fel de rădăcină a Registry-ului concepută special pentru *PowerShell* şi denumită `REGISTRY::`, ca în exemplul următor:

```
PS HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run> cd registry::
PS Microsoft.PowerShell.Core\Registry::> ls


   Hive:

SKC  VC Name                           Property
---  -- ----                           --------
  7   0 HKEY_LOCAL_MACHINE             {}
 14   0 HKEY_CURRENT_USER              {}
373   0 HKEY_CLASSES_ROOT              {}
  2   0 HKEY_CURRENT_CONFIG            {}
  0   2 HKEY_PERFORMANCE_DATA          {Global, Costly}
  6   0 HKEY_USERS                     {}
```

0-10: *Hive*-urile Registry-ului văzute prin intermediul *PowerShell Registry Provider*

Un exemplu de utilizare al celorlaltor *hive*-uri este următoarea secvenţă de cod care afişează aplicaţia asociată cu tratarea unei anumite extensii, în exemplul de mai jos, `.mp3`:

```
PS HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run> cd registry::
PS Microsoft.PowerShell.Core\Registry::> cd HKEY_CLASSES_ROOT\.mp3
PS Microsoft.PowerShell.Core\Registry::HKEY_CLASSES_ROOT\.mp3> Get-ItemProperty .

[...]
PerceivedType : audio
(default)     : Winamp.File.MP3
Content Type  : audio/mpeg
Winamp_Back   : WMP11.AssocFile.MP3
```

0-11: Afişarea aplicaţiei care tratează o anumită extensie

0.3.2.14 Ora şi data

Calcularea de intervale de timp, selecţia fişierelor pe baza unui interval, ordonarea cronologică, precum şi multe alte utilizări ale informaţiilor legate de timp sunt deseori utile în scripturile folosite de administratorii de sistem.

Principala comandă *PowerShell* pentru aflarea timpului este `Get-Date`[1]:

```
PS C:\Users\Administrator> Get-Date

Saturday, August 02, 2008 7:49:00 PM
```

Pentru a separa data şi ora curentă, se foloseşte parametrul `DisplayHint` lângă care se poate specifica `date` sau `time`:

```
PS C:\Users\Administrator> Get-Date -DisplayHint date

Saturday, August 02, 2008

PS C:\Users\Administrator> Get-Date -DisplayHint time

7:49:00 PM
```

[1] De fapt, `Get-Date` returnează deopotrivă informaţii legate de data şi ora curentă. Nu există o comandă `Get-Time` ci doar parametri suplimentari ai lui `Get-Date`.

Material cu drept de autor

Există și metoda `IsDaylightSavingTime()` care returnează `True` sau `False` în funcție de prezență orei de vară. Rezultatul asupra căruia se aplică metoda este cel al comenzii `Get-Date`, care trebuie încadrată între paranteze înainte de a i se accesa metodele:

```
PS C:\Users\Administrator> (Get-Date).IsDaylightSavingTime()
True
```

O utilizare frecventă a datelor în *scripting* o reprezintă adăugarea unei astfel de informații în numele fișierelor, menținându-le astfel ordonate. Următoarea secvență arată modul în care se pot obține noi nume pentru fișiere, concatenând data curentă la numele lor:

```
$filename = "fisier"
$datestring = Get-Date -uformat %Y%M%d
$newfilename   $filename + "_" + $datestring + ".txt"
Write-Host $newfilename
```

De remarcat faptul că, în linia de formatare a datei, parametrul `uformat` provine de la „UNIX format". Pentru exemplul de mai sus, `%Y` reprezintă un an din patru cifre, ca 2008, `%M` reprezintă o lună din două cifre, ca 08, iar `%d` o zi de două cifre, ca 02.

O listă parțială de modificatori acceptați la formatarea datei și a orei este următoarea:

- `C` Impropriu denumit secol (century), de fapt extrage primele două cifre ale anului (20 pentru 2008)
- `Y` An de patru cifre
- `b` Numele prescurtat al lunii
- `B` Numele complet al lunii
- `M` Lună de două cifre
- `V` Numărul săptămânii în intervalul 01..53
- `a` Numele prescurtat al zilei săptămânii
- `A` Numele complet al zilei săptămânii
- `u` Numărul zilei săptămânii, începând cu 1 corespunzător lui "luni" (*Monday*)
- `d` Ziua din lună în două cifre
- `j` Ziua din an
- `r` Timpul în format de 12 ore
- `T` Timpul în format de 24 de ore
- `p` ante meridian (a.m.) sau post meridian (p.m.)
- `H` Ora în format de 24 de ore
- `I` Ora în format de 12 ore
- `m` Minute
- `S` Secunde

Calcularea datelor din trecut sau viitor se face tot prin metode aplicate lui `Get-Date`, ca în exemplele de mai jos. Se observă că scăderea unităților de timp se face prin aceleași metode, dar argumente negative:

```
(Get Date).AddMonths(3)
(Get Date).AddHours(145)
(Get-Date).AddYears(-2)
```

Setarea datei se face prin comanda `Set-Date` urmată de parametrul `date` și un șir ce descrie noua valoare. Poate fi folosită și în combinație cu calcularea timpului în trecut sau viitor:

```
Set-Date -date "8/7/2008 12:00 AM"
Set Date (Get Date).AddHours(3)
```

Calcularea intervalului de timp dintre două evenimente este util în special pentru măsurarea timpului de rulare a unui script și poate fi realizată ca în exemplul de mai jos:

```
$start   Get-Date
```

Material cu drept de autor

```
<instrucțiuni>
$stop = Get-Date
$timediff = New-TimeSpan $start $stop
Write-Host "Scriptul a rulat " + $timediff.milliseconds + " millisecunde."
```

0-12: Calcularea timpului de rulare a unui script

Bineînțeles, în exemplul de mai sus se pot folosi și zile, ore, minute, secunde, *ticks* precum și alte proprietăți în locul milisecundelor.

Pentru a calcula prin intermediul unui *TimeSpan* timpul dintre momentul curent și o dată oarecare din prezent sau trecut, se poate folosi o comandă cu sintaxa următoare:

```
New-Timespan $(Get-Date) $(Get-Date -month 1 -day 1 -year 2027)
```

Comanda `Get-Date` poate returna obiecte cu proprietăți explicit specificate prin intermediul parametrilor ca `month`, `day` sau `year`. Utilizarea parantezelor este necesară în exemplul de mai sus deoarece se dorește evaluarea lor înaintea transmiterii rezultatelor ca parametrii pentru comanda `New-TimeSpan`.

Material cu drept de autor

1 Nivelul fizic

Ce se învață din acest capitol:

- Tipuri de semnale folosite în rețele de calculatoare
- Medii de transmisie
- Echipamente de rețea

Cine este...

Alexander Graham Bell *este un inventator și om de știință eminent, creditat cu inventarea telefonului. În 1881, primele cabluri torsadate (twisted pair) au fost folosite în sistemele telefonice proiectate de Bell.*

1.1 Semnale

La nivel fizic, unitatea de organizare a datelor este bitul. Biții pot fi reprezentați și transmiși printr-un canal de comunicație cu ajutorul semnalelor. Rolul fundamental al nivelului fizic este acela de a transforma semnalul în biți. Știind că la nivel fizic se lucrează doar cu semnale, sunt importante mijloacele prin care biții pot fi transportați cât mai eficient.

În cele ce urmează vor fi prezentate diverse tehnici și dispozitive ce fac posibilă optimizarea transferului de biți printr-un canal fizic.

1.1.1 Tipuri de semnale

1.1.1.1 Semnale analogice şi digitale

La nivelul cel mai general, un semnal este un fenomen fizic măsurabil, care variază în spațiu și/sau timp, utilizat pentru a transmite informație. Semnalele pot fi continue sau discrete. De asemenea, o clasificare frecventă a semnalelor este cea analogic/digital. Semnalele digitale sunt discrete și cuantizate – adică pot fi reprezentate prin numere cu un anumit nivel de precizie prestabilit. Semnalele analogice sunt continue și, teoretic, ar putea fi reprezentate prin numere doar cu un nivel infinit de precizie (cu un număr infinit de zecimale).

Semnalele analogice, cel mai adesea, sunt cele întâlnite în natură, cum ar fi vocea umană, ciripitul păsărilor, șuieratul vântului etc. Atunci când sunt reprezentate grafic ele seamănă cu niște valuri mai mult sau mai puțin simetrice. Cel mai simplu exemplu de semnal analogic este o sinusoidă (Figura **Error! Reference source not found.**). Semnalele analogice variază continuu în timp și de aceea nu au treceri bruște de la o valoare la alta: se mai spune că sunt *„wavy"*, adică unduioase .

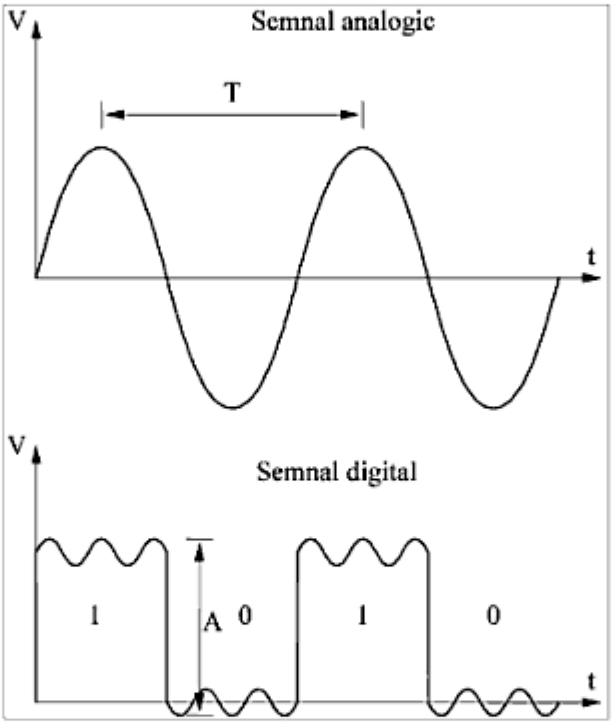
Semnalele digitale, cel mai adesea, sunt cele folosite în tehnică și au la bază două valori logice, 0 și 1, care au fiecare câte o reprezentare în funcție de modul în care sunt transmise. Impulsurile digitale (0 sau 1 logic) se numesc biți. Transmisia digitală este de multe ori de preferat celei analogice deoarece este mai puțin afectată de zgomote, fiind deci mai robustă. Datorită trecerilor bruște de la o valoare la alta, se mai spune că este *„jumpy"*, adică săltăreață. Semnalele digitale mențin un nivel constant de tensiune sau intensitate luminoasă, apoi trec pe alt nivel constant.

Exemplul simplu de mai jos ilustrează faptul că transmisia digitală e mai puțin afectată de zgomote. Fie o linie pe care se dorește transmiterea numărului 7. Dacă transmisia este analogică, se va transmite practic o undă, în exemplu de față amplitudinea fiind de 0,7 (dacă s-ar fi dorit transmiterea numărului 5, ar fi trebuit folosită o amplitudine 0,5, etc). Dacă acea linie este afectată de interferențe electromagnetice cu amplitudinea de 0,2, atunci la recepție

Material cu drept de autor

se va citi 0,9, adică numărul 9. Transmisie eronată! Dacă în schimb se foloseşte transmisia digitală, va trebui convertit 7 în binar, iar numărul 111 va fi transmis digital. Transmisia poate avea 2 valori: spre exemplu, 0 logic între amplitudinile 0,1 şi 0,4 şi 1 logic între 0,8 şi 1. Dacă se doreşte transmiterea lui 1 logic de 3 ori, practic vor fi transmise 3 impulsuri cu amplitudinea de 0,8. Dacă la ele se adaugă interferenţele prezente pe linie, la celălalt capăt vor fi citite 3 impulsuri de amplitudine 1, ceea ce înseamnă tot 1 logic. Transmisie corectă! Este adevărat că există numeroase cazuri în care datorită interferenţelor prea mari se emite 0 şi se recepţionează 1 sau invers, însă, în comparaţie cu transmisia analogică, cea digitală este mult mai precisă şi mai robustă.

Tipurile de semnale discutate mai jos sunt în marea lor majoritate digitale (acestea sunt cele mai folosite în reţele de calculatoare); atunci când este vorba despre semnale analogice, acest lucru va fi precizat explicit.



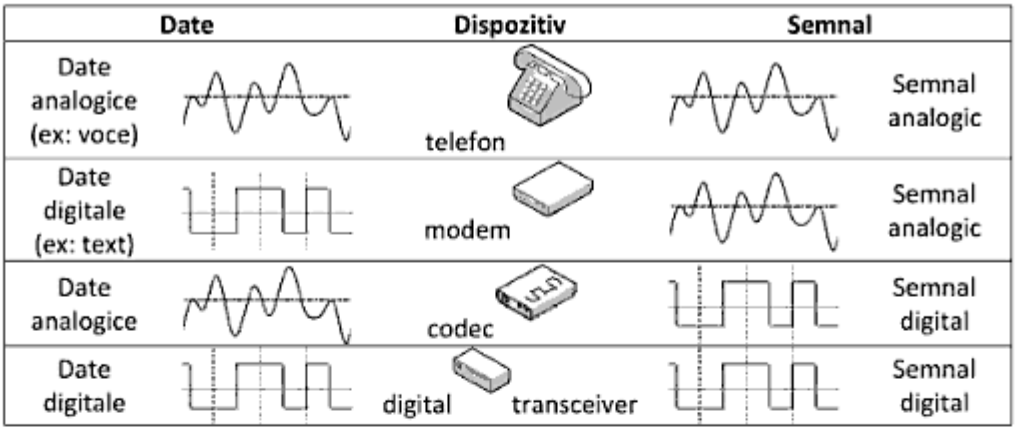
1-1: Exemplu de semnal analogic şi digital

Trebuie făcută distincţia între tipul semnalului şi tipul datelor transmise folosind acel semnal. La rândul lor, şi datele se împart în analogice sau digitale. **Datele analogice** sunt valori continue din cadrul unui interval (exemplu: sunetele din natură, înălţimea unei coloane de mercur din termometru). **Datele digitale** sunt valori discrete (exemplu: un fişier text, cifrele afişate pe ecranul unui termometru digital).

Un caz în care date analogice, cum ar fi vocea, sunt transmise printr-un semnal analogic, este cel al telefonului clasic.

De obicei, pentru datele digitale se folosesc semnale digitale. Dacă însă, se doreşte transmiterea de date digitale printr-un mediu analogic, trebuie folosit un modem. Acesta preia datele digitale de transmis şi le modulează, rezultând un semnal analogic. La recepţie, aplicând procesul invers, demodularea, asupra semnalului analogic citit de pe mediu se obţin datele digitale.

Material cu drept de autor



1-2: Date analogice/digitale, semnal analogic/digital

Tehnologia *Voice over IP* permite transferul datelor analogice prin semnale digitale. Codificarea datelor se poate face *software* sau direct *hardware*.

1.1.1.2 Clasificarea semnalelor în funcție de modul de transmisie

În funcție de natura generatorului de semnal şi a mediului în care se propagă, semnalele pot fi împărțite în trei categorii:

1.1.1.2.1 Semnale electrice

Semnalele electrice constau în impulsuri electrice ce folosesc ca suport pentru transmisie fire de cupru.

1.1.1.2.2 Semnale optice

Semnalele optice se obțin prin conversia semnalului electric în impulsuri luminoase care sunt transmise apoi printr-o fibră optică.

1.1.1.2.3 Unde electromagnetice (unde radio, microunde)

Semnalele *wireless* (fără fir) se propagă prin aer, sub formă de unde radio sau microunde.

1.1.2 Codarea

De-a lungul timpului au existat numeroase forme de transport al informației pe distanțe lungi. Fiecare dintre aceste metode avea o anumită formă de codare a informației. De exemplu, indienii *apache* făceau un foc mare pe un deal şi cu ajutorul unei pături formau rotocoale de fum. O variantă de codare folosită ar putea fi: 3 rotocoale de fum înseamnă că este vânat mult prin zonă, 4 rotocoale mari şi două mici înseamnă că vine furtuna, etc. Apariția codului *Morse* a revoluționat la vremea respectivă comunicațiile: fiecare literă avea propriul ei simbol format din semnale lungi şi scurte.

Procesul de transformare a informației într-un semnal ce poate fi transportat pe un canal fizic se numește ***codare.***

Transmiterea informației în rețelele de calculatoare presupune aplicarea mai multor procese de codare la diferite niveluri ale stivei OSI – precum segmentarea datelor, comprimarea datelor sau criptarea. Desigur, pentru a transmite informația, aceasta trebuie

Material cu drept de autor

convertită într-un semnal digital binar. La nivelul fizic, pasul următor constă în codarea semnalului binar într-un alt semnal adecvat mediului fizic – precum variații ale nivelului de tensiune într-un cablu de cupru, sau variații ale luminozității într-o fibră optică. Mai jos sunt prezentate câteva metode de codare ale semnalelor binare în semnale fizice.

1.1.2.1 Sincronizarea cu ceas

1.1.2.1.1 NRZ-L

În codarea *Non-Return-to-Zero Level* valoarea 1 logic este transmisă ca o tensiune joasă (de obicei negativă, de exemplu între -12V și -5V) iar 0 logic ca un nivel de tensiune înaltă (pozitivă, de exemplu între 5V și 12V). În reprezentarea unui șir de biți nivelul semnalului urmărește starea bitului.

Un dezavantaj important al acestei metode de codare este riscul crescut de pierdere a sincronizării la receptor. Transmiterea unei secvențe de date ce conține un număr mare de biți consecutivi cu aceeași valoare presupune menținerea tensiunii mai mult timp pe același nivel, iar în cazul desincronizării, numărul biților recepționați poate fi eronat.

1.1.2.1.2 NRZI

În codarea *Non-Return-to-Zero Inverted* valoarea semnalului trece de pe un nivel pe altul doar atunci când în șirul de biți apare valoarea 1 logic. Ca exemplu, dacă în starea curentă semnalul se afla pe nivelul de tensiune joasă, la apariția unui bit de valoare 1, va trece pe tensiune înaltă. Apariția unuia sau mai multor biți de 0 nu schimbă în niciun fel nivelul de tensiune. Acesta va reveni la tensiune joasă doar pentru a reprezenta următorul bit de 1 întâlnit în șir.

1.1.2.2 Sincronizarea fără ceas

1.1.2.2.1 Manchester

Codarea *Manchester* folosește pentru reprezentarea fiecăreia dintre cele două valori logice câte o tranziție între nivelurile de tensiune. Astfel, o trecere sus-jos codifică un bit 0, în timp ce un bit 1 este codificat printr-o trecere jos-sus. Tranzițiile au loc la mijlocul celulei de bit, ceea ce înseamnă că, dacă se pierde sincronizarea, pot fi folosite atât ca date cât și ca semnal de ceas. De exemplu, dacă este folosită codarea NRZ-L și trebuie transmiși 20 de biți de 1 logic, atunci ar fi necesare 20 de impulsuri de tensiune -5V. S-ar putea însă ca la recepție, datorită tuturor fenomenelor discutate până acum, să fie citiți 18 biți sau 21 de biți. Folosind codarea *Manchester*, unde fiecare bit e o tranziție, sunt trimise practic mai multe impulsuri electrice, însă la recepție vor fi citite tot 20 de tranziții.

Codarea *Manchester* este utilizată în cadrul standardului IEEE 802.3 (*Ethernet*).

1.1.2.2.2 Manchester diferențial

Manchester diferențial este o metodă de codare în care datele sunt combinate cu semnalele de ceas pentru a forma un șir de date cu autosincronizare. Această metodă folosește tranziția din mijlocul celulei de bit doar ca și semnal de ceas. Pentru a reprezenta 1, prima jumătate a celulei de bit curente este egală cu ultima jumătate a bitului precedent. Pentru a codifica 0, se inversează nivelul de tensiune existent în cea de-a doua jumătate a semnalului anterior. Cu alte cuvinte, un bit 0 este reprezentat printr-o tranziție la începutul celulei de bit, absența acestei tranziții semnificând 1 logic.

Manchester diferențial este utilizat în cadrul standardului 802.5 (*Token Ring*).

Material cu drept de autor

Pentru exemplificare, în figura 1-3 este reprezentată codarea caracterului K în cele patru variante discutate.

Caracterul A în hexazecimal are valoarea 0x41. Cum litera K se află la o distanță de 10 litere de A, înseamnă că reprezentarea lui K în hexa este 0x4B.

A = 0x41, B = 0x42, ..., I = 0x49, J = 0x4A,

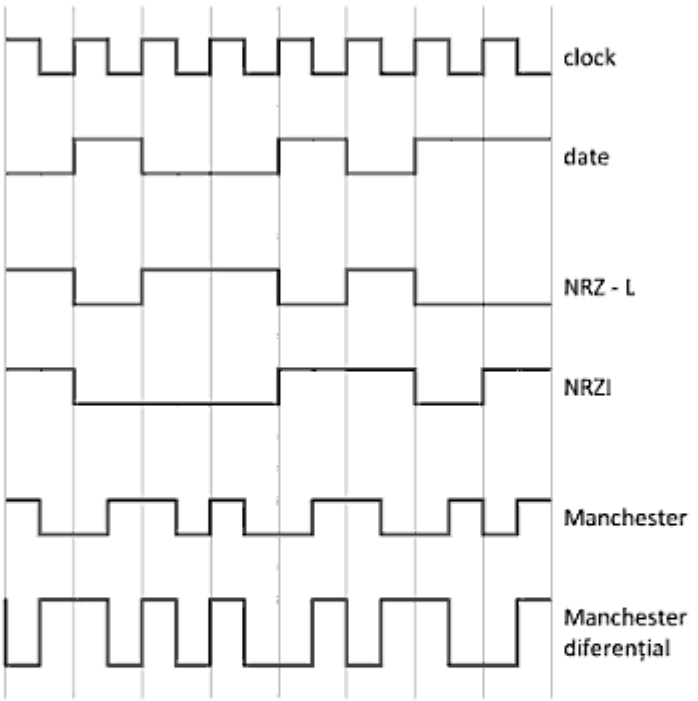
K = 0x4B

Reprezentarea binară:

$4_{(16)} = 4_{(10)} = 0100_{(2)}$

$B_{(16)} = 11_{(10)} = 1011_{(2)}$

Reprezentarea binară pentru litera K este 0100 1011.



1-3: Metode de codare

1. Codarea NRZ-L - dacă un bit este 1, semnalul este pe nivelul de tensiune joasă, dacă bitul este 0, semnalul trece pe tensiune înaltă
2. CodareaNRZ-I - semnalul schimbă nivelul de tensiune doar când urmează un bit 1.
3. Codarea *Manchester* – 0 este codificat ca o tranziție sus-jos, 1 ca tranziție jos-sus
4. Codarea *Manchester* diferențial - Tranziția de la începutul semnalului indică un bit 0.

1.1.3 Modularea

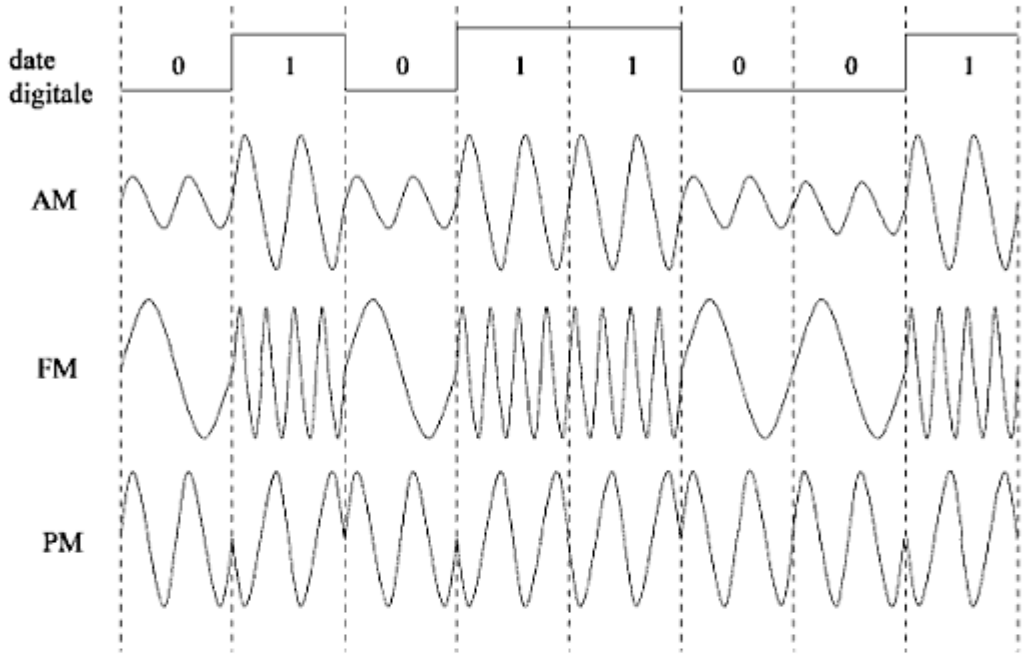
Modularea se referă la modificarea unui semnal folosind un alt semnal. Într-o transmisie radio semnalul cu ajutorul căruia este transportată informația este o undă, de exemplu o sinusoidă. Transmițătorul emite în permanență o sinusoidă (caracterizată de amplitudine, frecvență şi fază) cu toți parametrii constanți. În acest caz, cantitatea de informație este nulă, adică pe această sinusoidă nu este transmisă niciun fel de informație utilă. În momentul în care începe transmisia datelor, semnalul util de date, adică biții, sunt folosiți pentru a varia parametrii sinusoidei. Cu alte cuvinte, datele - adică biții – se reprezintă prin modificări ale

Material cu drept de autor

sinusoidei inițiale. Cum, sau mai exact, ce trebuie modificat la sinusoidă? Pot fi modificați următorii parametri:

- amplitudinea: modulare AM - *amplitude modulation*;
- frecvența: modulare FM - *frequency modulation*;
- faza: modulare PM - *phase modulation*.

***Modularea** este procesul de compunere a unei unde purtătoare cu un set de date.*



1-4: Modulare în amplitudine (AM), frecvență (FM), fază (PM)

Desigur, există forme mult mai avansate de modulare, însă cele trei prezentate mai sus reprezintă bazele modulării semnalelor.

Datele din calculator fiind digitale, pentru orice fel de comunicație trec prin procesul de codare. Dacă mediul de transmisie folosit este tot digital (de exemplu cablu UTP), datele sunt puse direct pe mediu, fără a mai fi nevoie de modulare. Pentru transmisiile pe legături seriale sau pe cablu coaxial va fi folosită atât codarea cât și modularea.

1.1.4 Multiplexarea

Multiplexarea este procedeul prin care mai multe canale de date sunt combinate într-un singur canal fizic. Demultiplexarea este procesul invers multiplexării, de separare a canalelor inițialedin canalul fizic .

Există numeroase tehnici de multiplexare, între care se numără:

- TDM (*Time Division Multiplexing*): informațiilor din fiecare canal de date li se alocă o cuantă de timp predefinită, indiferent dacă pe acele canale se transmite sau nu.
- ATDM (*Asynchronous time-division multiplexing*): informațiilor din fiecare canal de date li se alocă o cuantă de timp variabilă, în funcție de numărul de canale utilizate în acel moment.
- FDM (*Frequency Division Multiplexing*): fiecare canal primește o anumită bandă de frecvență.
- *Statistical multiplexing* - Banda este alocată în mod dinamic fiecărui canal care are informații de transmis.
- DWDM (*Dense Wavelength Division Multiplexing*) este o formă de multiplexare dezvoltată pentru transmisia pe fibră optică. DWDM este echivalentul optic al multiplexării FDM.

Material cu drept de autor

Aceste tipuri de multiplexări se referă la mărimea fizică ce stă la baza separaţiei canalelor. De exemplu, în cazul multiplexării TDM, fiecărui canal de comunicaţie i se alocă o cuantă de timp, iar în cazul FDM, fiecărui canal i se alocă o anumită bandă de frecvenţă.

1.1.5 Caracteristici ale semnalului

1.1.5.1 Latenţa

Latenţa, numită şi întârziere, este de două tipuri: latenţa propagării prin mediul de transmisie şi latenţa trecerii prin echipamentele de reţea.

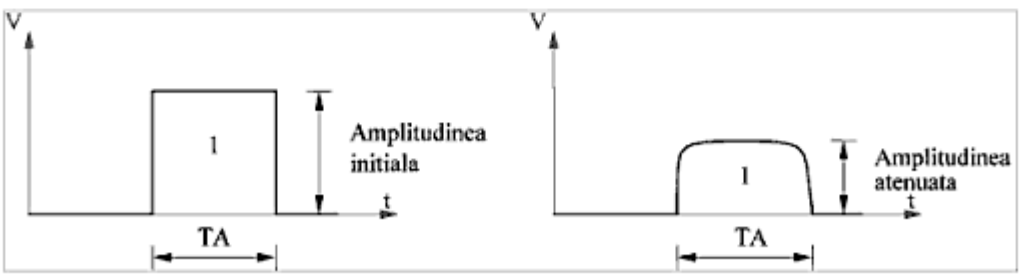
Primul tip de latenţă este dat de viteza de propagare a semnalului în mediul de transmisie specific şi de distanţa între sursă şi destinaţie. De exemplu, pentru o transmisie prin mediul electric viteza de propagare a semnalului este aproximativ două treimi din viteza luminii. Aceasta înseamnă că un impuls electric va parcurge un segment de reţea de 100 m în $\frac{100}{(\frac{2}{3}\cdot 3\cdot 10^8)} = 0{,}5\cdot 10^{-6}$ secunde.

A doua sursă a latenţei o reprezintă echipamentele de reţea folosite pe parcurs. Fiecare echipament execută operaţii specifice, de la redresarea semnalului electric, până la determinarea căii optime pe care trebuie trimis fiecare pachet. Latenţa dispozitivelor de interconectare variază de la câteva microsecunde în cazul hubului şi a convertoarelor de mediu, până la milisecunde în cazul comutatoarelor şi a routerelor. Astfel, comparativ cu latenţa introdusă de un repetor *Ethernet*, de aproximativ 5,6 microsecunde, latenţa mediului de conectare este cu un ordin de mărime mai mică.

Latenţa propagării este în general semnificativ mai mică decât latenţa dispozitivelor de interconectare, astfel încât deseori este considerată drept neglijabilă. Cu toate acestea, există cazuri în care latenţa propagării este factorul principal al întârzierii totale a unui semnal, cel mai relevant exemplu fiind cel al comunicaţiilor prin satelit. Folosirea sateliţilor geostaţionari face ca drumul total între sursă şi destinaţie să fie de peste 75.000 km, aducând latenţa totală a oricărei transmisiuni în jurul valorii de 0,5 secunde.

1.1.5.2 Atenuarea

„Atenuarea" este un termen general care se referă la reducerea puterii unui semnal. Atenuarea are loc indiferent de tipul de semnal, analogic sau digital. Numită uneori şi „pierdere" (*loss*), atenuarea este o consecinţă a transmiterii semnalului la distanţe mari. Atenuarea afectează reţelele de calculatoare deoarece limitează distanţa maximă între dispozitivele acesteia. Dacă distanţa este prea mare, din cauza atenuării, la destinaţie nu se va mai putea interpreta semnalul corect.



1-5: Atenuarea semnalului

Material cu drept de autor

Pentru transmisia la distanţe mai mari decât permite tipul de cablu utilizat se folosesc anumite dispozitive, numite **repetoare**, care regenerează semnalul (din punct de vedere electric, optic sau *wireless*). Atenuarea afectează toate tipurile de medii de transmisie, însă are valori diferite pentru fiecare mediu în parte. De exemplu, un semnal electric transmis pe un fir de cupru se atenuează mai repede decât un semnal optic (transmis pe o fibră optică). Atenuarea în general se măsoară în decibeli (dB), iar atenuarea specifică unui anumit tip de cablu se măsoară în decibeli/metru sau decibeli/kilometru. Fiecare tip de cablu are o atenuare specifică. Cu cât această atenuare este mai mică, cu atât acel cablu este considerat mai bun. Atenuarea este un factor foarte important de luat în calcul în cazul proiectării reţelelor de fibră optică. Echipamentele de fibră optică garantează o anumită distanţă (specificată în cartea tehnică), însă această distanţă este garantată pentru o fibră optică cu o anumită atenuare / km (specificată tot în cartea tehnică). Dacă se foloseşte o fibră optică cu o atenuare mai mare, atunci distanţa maximă garantată va fi mai mică. Dacă însă se foloseşte fibră optică de o mai bună calitate, transmisia va fi corectă şi la distanţe mai mari decât cea specificată.

Cum se determină distanţa maximă posibilă pentru o transmisie?

Echipamentele impun o anumită valoare a atenuării care nu trebuie depăşită. Se poate considera că:

$$distanţa\ maximă = \frac{atenuarea\ maximă\ a\ echipamentului - atenuarea\ conectorilor}{atenuarea\ specifică\ a\ mediului}$$

1.1.5.3 Reflexia

Reflexia are loc de obicei atunci când un semnal întâlneşte o linie de separaţie între două medii. Atunci, o anumită parte din semnal se reflectă înapoi în mediul din care a venit şi o parte trece în mediul următor.

Reflexia poate apărea în cazul semnalelor electrice când, de exemplu, impulsurile electrice sau biţii întâlnesc o discontinuitate, moment în care o anumită parte din energia semnalului se reflectă. Dacă nu este controlată, această energie poate interfera cu biţii transmişi mai târziu. Milioane de biţi sunt transmişi în fiecare secundă, iar această energie reflectată poate duce la multe transmisii nereuşite. Un exemplu este o reţea pe cablu coaxial care are nevoie de un terminator la fiecare capăt. Dacă nu ar avea acest terminator, la capătul cablului ar apărea o linie de separare între cele două medii (aer şi cupru), iar o parte din energie s-ar reflecta înapoi în firul de cupru.

Reflexia poate avea loc şi în cazul sistemelor optice. Un semnal optic se reflectă ori de câte ori întâlneşte o discontinuitate în fibra de sticlă, ca de exemplu atunci când se ataşează un conector. De aceea este necesară o pregătire specială în cazul ataşării conectorilor de fibră optică, pentru a nu permite reflexia luminii înapoi în fibră.

1.1.5.4 Zgomotul

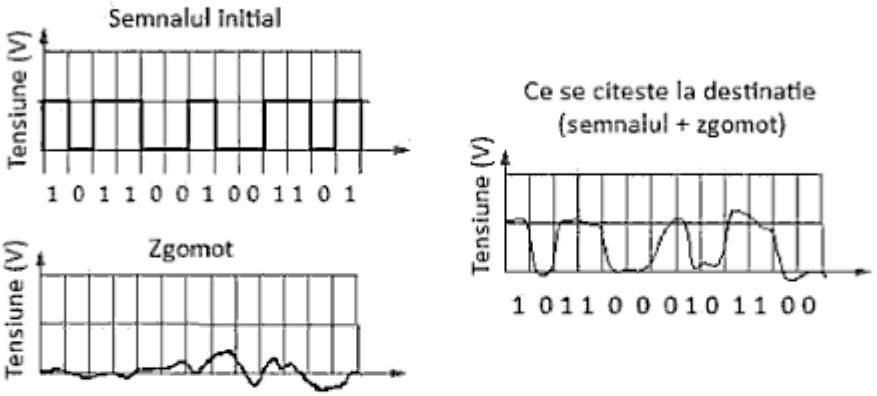
Zgomotul este o cantitate de energie nedorită (electrică, electromagnetică sau radio) care poate degrada calitatea semnalului transmis. Zgomotul afectează atât transmisiile analogice cât şi cele digitale. În cazul semnalelor analogice, semnalul devine bruiat şi uşor deformat. Un exemplu este o convorbire telefonică pe care se aude un zgomot de fond. În sistemele digitale, zgomotele afectează valorile biţilor transmişi (0 sau 1), la destinaţie aceştia putând fi interpretaţi greşit (adică 1 în loc de 0 şi invers).

Zgomotul poate avea mai multe cauze: câmpurile electrice provenite de la motoare electrice, lumina fluorescentă (neoane), etc. - toate provenite de la surse exterioare cablului afectat. Acest tip de zgomot se numeşte EMI (*Electromagnetic Interference* - Interferenţă

Material cu drept de autor

Electromagnetică) dacă provine de la surse electrice sau RFI (*Radio Frequency Interference* - Interferență Radio) când provine de la surse radio, radar sau microunde. Zgomotul mai poate proveni de la liniile de curent alternativ sau de la fulgere.

Fiecare fir dintr-un cablu poate acționa ca o antenă. Când acest lucru se întâmplă, firul practic absoarbe semnale electrice din celelalte fire din cablu sau din surse electrice exterioare cablului. Dacă zgomotul electric rezultat atinge un nivel destul de înalt, poate deveni foarte dificil sau chiar imposibil pentru echipamentul de la celălalt capăt să distingă semnalul de zgomot.



1-6: Efectul zgomotului

Un sistem de transmisie poate fi afectat de unele dintre aceste tipuri de zgomot şi imun la altele. De exemplu, transmisia optică este imună la interferențele electrice, deoarece semnalul purtat nu are natură electrică, ci optică. Acest lucru le face ideale pentru legăturile din exteriorul clădirii, unde transmisia pe firele de cupru ar putea fi influențată de fulgere, câmpuri electrice din alte surse, etc.

1.1.5.5 Crosstalk

Cablurile de cupru sunt afectate de interferențe electromagnetice de la diferite surse din afara cablului. Totuşi, cea mai importantă sursă de zgomot pentru cablurile de cupru o reprezintă efectul numit **crosstalk**: interferența semnalelor între două fire din interiorul aceluiaşi cablu. Una dintre cele mai eficiente metode de prevenire a efectului de *crosstalk* este torsadarea firelor. Prin torsadare, câmpurile electrice se anulează şi firele din celelalte perechi nu mai sunt influențate de semnalul din perechea inițială. De multe ori apar însă probleme la atașarea conectorilor. După cum se va vedea în studiul de caz din acest capitol, atunci când se dorește atașarea unui conector la capătul unui cablu trebuie întâi detorsadate toate perechile din interiorul cablului. Dacă se lasă o bucată prea mare detorsadată, în acea zonă câmpurile electrice generate de fiecare fir dintr-o pereche nu se vor mai anula şi va apărea o interferență între fire, numită NEXT (*Near-End Crosstalk*). Acest parametru, NEXT, este specific fiecărui cablu. Cu cât un cablu este terminat (adică mufa este sertizată) cu mai multă atenție, cu atât efectul NEXT va fi mai mic. Valoarea maximă a parametrului NEXT este specifică fiecărei categorii de cablu (Cat3, Cat5, Cat6): cu cât categoria este mai mare, cu atât interferența NEXT trebuie să fie mai mică (adică se impune o calitate mai ridicată a sertizării cablurilor).

Terminarea cu grijă a cablurilor este cea mai importantă metodă de prevenire a efectului de *crosstalk*.

Material cu drept de autor

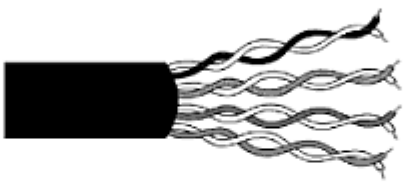
1.2 Soluții de comunicație pe cupru

1.2.1 Cablul coaxial

Rețele de cablu coaxial au avut perioada de impact maxim la jumătatea anilor `90. Odată cu apariția mediilor torsadate (UTP, STP) popularitatea lor a început să scadă. Deși oferă o mai bună ecranare și permit distanțe mai mari, mediul coaxial este unul analogic, spre deosebire de mediul torsadat unde transmisia se realizează digital. Eliminarea etapelor de conversie digital-analogic au permis costuri mai reduse pentru echipamentele de rețea destinate rețelor bazate pe UTP. În plus, folosirea unor perechi distincte pentru transmisie și recepție fac din UTP un mediu de comunicație *full-duplex*, spre deosebire de rețelele bazate pe medii de transmisie coaxiale. Rețelele de date bazate pe cablu coaxial mai pot fi încă întâlnite în cazul unor mici rețele de cartier, dar în ultimii ani acestea au devenit extrem de rare.

1.2.2 Cablul torsadat

Cablul torsadat este format din mai multe fire de cupru izolate, având o grosime tipică de 1mm, împletite două câte două (torsadate). Majoritatea cablurilor torsadate folosite pentru rețele locale conțin opt fire, așadar, patru perechi. Răsucirea firelor dintr-o pereche este necesară pentru anularea efectului de antenă caracteristic liinilor lungi. Acest efect ar produce interferențe electrice, ceea ce ar conduce la pierderi de date.



1-7: Cablu UTP

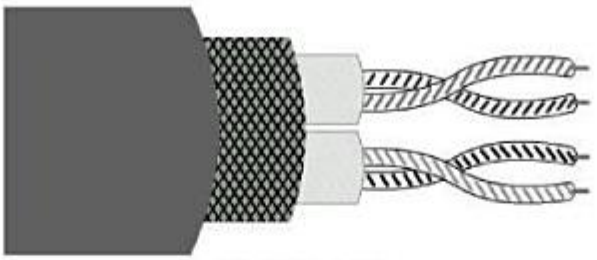
Pe lângă interferențele cauzate de câmpurile electrice induse de alte fire din interiorul aceluiași cablu, pot apărea și interferențe din surse exterioare cablului (de exemplu: existența unui motor electric în apropiere, sau, pentru cablurile aflate în exteriorul clădirilor, descărcările electrice din atmosferă). O metodă prin care se încearcă reducerea la minim a interferențelor exterioare este transmiterea diferențială. Transmiterea diferențială, sau transmiterea în mod balansat, presupune ca semnalul util transmis să reprezinte diferența dintre semnalele electrice de pe cele două fire ale unei perechi. Astfel, dacă apar interferențe electrice de la surse exterioare cablului, acestea vor afecta ambele fire în mod egal, diferența dintre semnale rămânând constantă. O altă metodă de prevenire a interferențelor exterioare este ecranarea cablurilor. Ecranarea presupune existența unui înveliș format dintr-o plasă sau o foiță metalică ce are rol de cușcă *Faraday*.

Din punct de vedere al ecranării, există două feluri de cabluri torsadate: ecranate (*shielded*) și neecranate (*unshielded*). Cele neecranate se numesc UTP (*unshielded twisted pair*) și sunt cele mai folosite în cadrul rețelelor locale de calculatoare, fiind, de altfel, și cele mai ieftine.

Dezavantajul cablurilor UTP este că nu pot fi folosite în exteriorul clădirilor, deoarece ar fi supuse unor posibile șocuri electrice foarte mari, ce ar duce la defectarea echipamentelor conectate. De aceea, în exteriorul clădirilor se folosește, în general, cablu ecranat: ScTP (*screened twisted pair*), STP (*shielded twisted pair*) sau S/STP (*screened shielded twisted pair*). ScTP, numit și FTP (*foiled twisted pair*), are un singur înveliș de ecranare exterior și este doar cu puțin mai gros decât UTP. Cablul STP are, pe lângă învelișul de ecranare identic cu cel de la

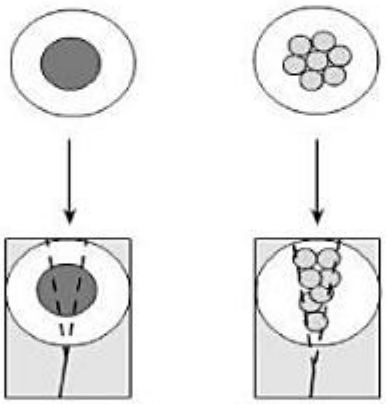
Material cu drept de autor

ScTP, câte un înveliş separat pentru fiecare pereche. Acest lucru îl face mult mai rezistent la interferenţe, dar şi mult mai scump. În plus, fiind mai rigid, este şi ceva mai greu de manevrat.



1-8: Cablu STP

Din punct de vedere al maleabilităţii, cablurile torsadate se împart în solide şi lităte. Cele solide au în interiorul fiecăruia dintre cele opt fire ale cablului câte un singur fir de cupru de aproximativ 1mm, spre deosebire de cele lităte, la care fiecare fir este format dintr-o mulţime de fire foarte subţiri, numite liţe. Cablurile lităte sunt aşadar mai flexibile, fiind potrivite pentru cablările orizontale (de la priza de perete până la staţia utilizatorului), în timp ce cablurile solide sunt folosite la cablările verticale (acolo unde este nevoie, de obicei, de cabluri rigide).



1-9: UPT solid şi lităt

1.2.2.1 Standarde pentru medii torsadate

Colecţia IEEE 802.3 cuprinde standardele ce definesc nivelul fizic şi subnivelul MAC al nivelului legătură de date pentru *Ethernet*. Este definit câte un standard pentru fiecare tip de mediu de transmisie folosit. Astfel, în această colecţie se regăsesc, printre altele, standardele pentru cablu UTP, standardele pentru *Ethernet* pe cablu coaxial (10BASE5, 10BASE2), *Ethernet* prin fibră optică (10BASE-F, 100BASE-FX, etc) sau descrierea tehnologiei PoE (*Power over Ethernet*).

Standardul ce conţie cerinţele pentru transmiterea a 10Mbit/s pe cablu UTP este standardul 10BASE-T. În mod similar, pentru 100 Mbit/s şi 1000 Mbit/s (1 Gbit/s) există 100BASE-T, respectiv 1000BASE-T (numit şi *Gigabit Ethernet*). Numele standardului derivă din unele aspecte legate de mediului fizic: Numărul reprezintă viteza maximă teoretică exprimată în megabiţi pe secundă. „BASE" este prescurtarea pentru *baseband*, ceea ce înseamnă că fiecare fir este folosit ca un singur canal de comunicaţie, pe care se transmite într-o singură

Material cu drept de autor

frecvență. Cu alte cuvinte, nu se aplică nicio formă de multiplexare. Litera de la sfârșit reprezintă tipul cablului, în acest caz, „T" înseamnă torsadat (*twisted*). Așadar, 100BASE-T este o denumire generică pentru un standard care asigură o viteză de 100Mbit/s pe cablu torsadat. În particular, sunt definite trei forme : 100BASE-TX, 100BASE-T4 și 100BASE-T2. 100BASE-TX indică utilizarea unui cablu de categorie cel puțin CAT5 și folosirea a 2 perechi de fire din cele 4. Sufixul T4 indică folosirea a 4 perechi pentru comunicație. 100BASE-T4 și 100BASE-T2 nu se mai folosesc, fiind standarde învechite. Toate aceste standarde operează pe segmente de cablu cu lungimi de maxim 100 de metri.

În 2006 a fost publicat standardul 10GBASE-T pentru conexiuni de 10 gigabit/s prin cablu torsadat. *10Gigabit Ethernet* suportă doar legături *full-duplex*, spre deosebire de celelalte trei standarde ce suportă și comunicații *half-duplex*.

După cum s-a menționat la începutul acestui capitol, cantitatea de informație transferată între emițător și receptor este proporțională cu frecvența semnalelor pe mediul de transmisie. În cazul semnalelor electrice, frecvența este dată de calitatea cuprului de a fi mai bun sau mai puțin bun conductor de curent electric. Această calitate depinde de densitatea de impurități caracteristică materialului. De aceea, există mai multe categorii de cabluri, o categorie mai mare implicând performanțe mai bune.

1.2.2.2 Categorii de medii torsadate

Categoriile de cabluri torsadate au fost definite în setul de standarde TIA/EIA-568-B de către asociația americană *Telecommunications Industry Association* (TIA). Acesta s-a dovedit a fi standardul cu cea mai largă acceptare în piața producătorilor de soluții pentru nivelul fizic.

1.2.2.2.1 UTP CAT1-4

Cablul încadrat la categoria 1 (CAT1) este cel folosit în serviciile de telefonie clasică (POTS – *Plain Old Telephone Service*) sau soneriile de la uși. Această etichetare este cumva improprie, întrucât setul de standarde TIA/EIA-568-B nu recunoaște în momentul de față decât categoriile 3, 5e, 6 și 6a.

Standardul C3 a fost folosit în anii `90 pentru *TokenRing* și pentru *Ethernet*, ajungând la viteze de până la 10Mbit/s. Astăzi, acesta este folosit în sistemele de telefonie și poate fi ușor adaptat pentru *Voice over IP* (VoIP) întrucât viteza de 10Mbit/s pe care o oferă depășește cu mult cerințele de 0,08Mbit/s ale unui telefon VoIP la încărcare maximă. În plus, CAT3 este compatibilă cu tehnologia *Power over Ethernet* (definită în standardul 802.3af PoE), tehnologie ce descrie un sistem prin care odată cu datele se transferă și energie electrică, tocmai în scopul alimentării anumitor aparate aflate la distanță, precum telefoanele VoIP. Apariția standardului 100BASE-T4 a dus la creșterea vitezei la 100Mbit/s prin utilizarea a 4 perechi de fire (și nu doar 2 cum prevedea standardul anterior), ceea ce a permis infrastructurilor mai vechi, deja existente, de cabluri CAT3 să ofere o lățime de bandă mai mare. Cu toate acestea, utilizarea sa pentru comunicațiile de date a scăzut odată cu apariția standardului CAT5.

Standardul CAT4 oferea o frecvență cu puțin mai mare decât CAT3, 20MHz față de 16MHz și era utilizat pentru o variantă îmbunătățită a rețelelor *Token Ring*.

1.2.2.2.2 UTP CAT5 și CAT5e

Specificațiile cablului de categoria 5, definite în TIA/EIA-568-B, indică o frecvență maximă de 100MHz. CAT5 este folosit în special în rețele de 100Mbit/s (*FastEthernet*), dar poate fi utilizat și pentru *Gigabit Ethernet*.

Odată cu definirea în 2001 a CAT5e (*enhanced*) în TIA/EIA-568-B, specificațiile variantei originale CAT5 nu mai sunt recunoscute în aceste standarde.

Material cu drept de autor

UTP CAT5e a devenit cel mai răspândit mediu de transmisie pentru rețelele locale. Datorită performanțelor îmbunătățite față de versiunea originală, și datorită unui preț mult mai mic decât al CAT6, CAT5e este cea mai potrivită alegere pentru infrastructura rețelelor *Gigabit Ethernet*. Cu toate acestea, CAT5e menține recomandarea limitării segmentelor de la cablu la 100 de metri, la fel ca și în cazul celorlalte tipuri de cabluri definite de TIA/EIA.

Este de reținut faptul că standardul folosit pentru *Gigabit Ethernet*, 1000BASE-T, impune utilizarea a 4 perechi de fire torsadate, spre deosebire de versiunile anterioare (10BASE-T și 100BASE-T) care foloseau în comunicație doar două perechi. Așadar, standardul de *Ethernet* ales pentru infrastructură este cel care specifică numărul de perechi necesare în comunicație, și nu standardul de cablu. Categoria specifică doar caracteristicile specifice cablului, precum: numărul de perechi existente, pasul de torsadare, diametrul firelor, parametrii NEXT, FEXT și, cel mai important, limita superioară de frecvență. Astfel, un cablu CAT5e folosit pentru 100BASE-T (*FastEthernet*) utilizează în comunicație 2 perechi de fire din cele 4 disponibile, în timp ce același cablu pentru infrastructuri de 1000BASE-T (*Gigabit Ethernet*) necesită toate cele 4 perechi.

1.2.2.2.3 UTP CAT 6, CAT6a

UTP CAT6 aduce îmbunătățiri majore, precum impunerea unui pas de torsadare mult mai mic decât la CAT5 și o limită superioară de frecvență de 250MHz, fiind conceput special pentru rețelele *Gigabit Ethernet*. Standardul de cablu categoria 6 păstrează compatibilitatea cu standardele CAT5, CAT5e și CAT3.

Deși CAT6 este mai frecvent folosit în rețelele *Gigabit Ethernet*, specificațiile sale permit și implementarea standardului 10GBASE-T (apărut în 2006), dar numai pe segmente de 55 de metri. Pentru a face posibilă utilizarea standardului 10BASE-T pe lungimi de 100 de metri, se impune folosirea unui nou tip de cablu, definit ca standard TIA în februarie 2008, și anume categoria 6a.

Cablul UTP CAT6a (augmented) operează la frecvențe de până la 500MHz (dublu față de CAT6), fiind destinat infrastructurilor de 10GBASE-T (10 Gigabit Ethernet).

1.2.2.2.4 UTP CAT7, CAT8

Standardul de cablul categoria 7 (CAT7) are un pas de torsadare și mai mic decât CAT6 și, în combinație cu conectori de tip GG45, poate trata semnale cu banda de frecvență de până la 625MHz. În plus, fiecare dintre cele patru perechi de fire este ecranată individual (pe lângă învelișul exterior al cablului). Deși a fost creat pentru *10 Gigabit Ethernet*, cea mai folosită tehnologie pentru 10GBASE-T rămâne CAT6a.

Categoria 7 este și cea mai strictă în privința normelor de siguranță referitoare la comportamentul cablurilor în situații de incendiu: viteza de răspândire a focului, substanțe emanate, etc. Un exemplu care să justifice necesitatea unor astfel de reglementări este cel al cablurilor cu învelișul din PVC, foarte populare datorită pretului scăzut. În momentul în care iau foc, aceste cabluri degajă substanțe foarte toxice omului, fiind total nepotrivite pentru cablările orizontale.

UTP CAT8 este destinat infrastructurilor multimedia, un astfel de cablu putând transporta simultan oricare patru servicii de tip TV, video, satelit, audio, date, etc. Cablul UTP Cat 8 operează cu frecvențe de 1200MHz și poate ajunge la maxim 1400MHz.

Material cu drept de autor

Categorie cablu	Frecvență	Viteza de transmisie	Utilizare
Cat 1		1Mbps	Telefonia clasică
Cat 2		4Mbps	Transmisiuni seriale
Cat 3	16MHz	10 Mbps 100 Mbps	TokenRing 10BaseT 100BaseT4
Cat 4	20MHz	16 Mbps 100 Mbps	TokenRing 10BaseT 100BaseT4
Cat 5	100MHz	10 Mbps 100 Mbps	ATM, TokenRing, 10BaseT 100BaseTX
Cat 5e	155MHz	10 Mbps 100 Mbps 1 Gbps	10BaseT, 100BaseTX, 1000BaseT
Cat 6	250MHz	100Mbps 1 Gbps	100BaseTX 1000BaseT
Cat 6a	500MHz	10 Gbps	10GBaseT
Cat 7	625MHz	10 Gbps	10GbaseT
Cat 8	1200Mhz	10 Gbps	10GbaseT

1-10: Categorii de cablu

1.2.2.3 Tipuri de cabluri UTP

Procedura de fixare a firelor unui cablu într-un conector se numeşte sertizare. Standardul TIA/EIA-568B specifică două moduri în care pot fi ordonate firele la o terminaţie a cablului, secţiunea corespunzătoare fiind probabil şi cea mai cunoscută din întreaga documentaţie. Pentru a fi uşor identificate, cele opt fire sunt colorate diferit. Culorile folosite pentru cele patru perechi sunt: albastru, verde, portocaliu şi maro. Pentru a deosebi firele unei perechi, unul are învelişul de culoare uniformă, celălalt având doar o dungă din culoarea respectivă pe fond alb. Cele două moduri specificate de TIA/EIA-568-B pentru ordonare firelor se numesc T568A (standard folosit mai mult în Statele Unite) şi T568B (folosit în general în Europa).

Pin	T568 B	Pin	T568 A
1	Alb-portocaliu	1	Alb-Verde
2	Portocaliu	2	Verde
3	Alb-Verde	3	Alb-portocaliu
4	Albastru	4	Albastru
5	Alb-albastru	5	Alb-albastru
6	Verde	6	Portocaliu
7	Alb-maro	7	Alb-maro
8	Maro	8	Maro

1-11: Codurile culorilor în cablul UTP

Material cu drept de autor

După cum se știe, tehnologiile 100BaseTX și 10BaseT folosesc doar două perechi din cele patru: una pentru transmisie (Tx+ și Tx-) și una pentru recepție (Rx+ și Rx-). Conform standardelor de mai sus, acestea sunt portocaliu și verde (pinii 1,2,3 și 6). Atenție: firele de Tx precum și firele de Rx trebuie să facă parte din aceeași pereche! Se observă că prima pereche ajunge pe pinii 1 și 2 iar a doua pereche pe pinii 3 și 6.

În funcție de corespondența perechilor dintr-un capăt cu pinii de la celălalt capăt, cablurile se împart în trei categorii:

1.2.2.3.1 Straight-through

Cablul direct (*straight-through*) are ambele capete sertizate conform aceluiași standard (T568A - T568A în SUA, sau T568B - T568B în Europa). Se folosește atunci când se conectează o stație la un switch sau la un hub. Cele două capete având aceeași ordine a firelor, fiecare pin al conectorului dintr-un capăt comunică direct cu pin-ul corespunzător al conectorului de la celălalt capăt al cablului.

Atunci când se conectează o stație la un switch sau hub se folosește un cablu direct!

1.2.2.3.2 Crossover

Cablul *crossover* se folosește la conectarea a două calculatoare între ele, fără a mai folosi un switch sau un hub. Prin felul în care este construit acest cablu, pinul 1 de la un capăt va corespunde pinului 3 de la celălalt capăt, iar pinul 2 pinului 6. Aceasta înseamnă că datele transmise prin perechea Tx de la un capăt vor ajunge pe pinii de Rx de la conectorul opus. Astfel, două calculatoare pot transfera date direct între ele, fără a mai trece printr-un alt echipament, dacă plăcile lor de rețea sunt legate printr-un cablu *crossover*. Întrucât singura diferență dintre T568A și T568B este inversarea perechii portocalii cu perechea verde, un cablu crossover poate fi văzut ca având un conector sertizat conform T568A și pe celălalt conform T568B. Un astfel de cablu va funcționa pentru standardul 10BASE-T sau 10BASE-TX, unde se folosesc doar 2 perechi. Pentru 1000BASE-T (*Gigabit crossover*) însă, trebuie inversate și celelalte două perechi (albastru și maro), și, în plus, schimbate între ele firele fiecărei perechi (cea dungată cu cea uniformă).

Pentru a transfera date direct între două stații, se folosește un cablu crossover!

1.2.2.3.3 Rollover

Cablul de consolă (*rollover*) este folosit atunci când se dorește conectarea pe un port de consolă a unui router. Există mai multe variante de cabluri ce pot fi folosite pentru a face legătura între un PC și un port de consolă al unui router. Întotdeauna portul calculatorului pentru o astfel de legătură este unul serial (DB-9 sau DB-25). Portul de pe router poate fi DB-25 sau RJ-45. Astfel, se poate folosi un cablu ce are ca terminatori o mufă DB-9 și una RJ-45 sau un cablu *rollover* și un adaptor RJ45 – DB9 (sau RJ45 – DB25).

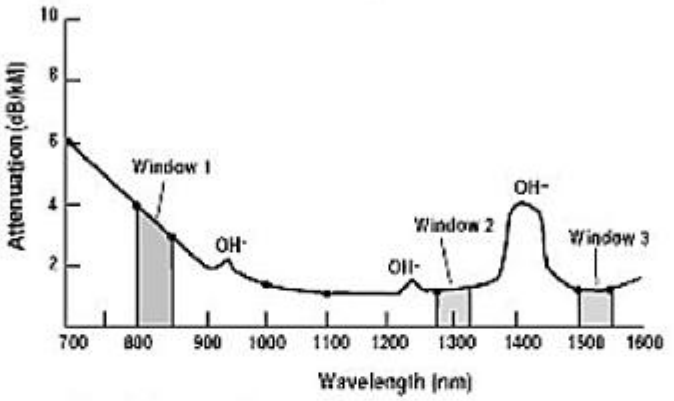
1.3 Soluții de comunicație pe fibră optică

Fibra optică este cel mai nou mediu de transmisie dezvoltat pentru rețele de calculatoare, având numeroase avantaje față de cablurile de cupru, dintre care cele mai importante sunt viteza de transmisie superioară pe care o suportă și imunitatea la interferențe electrice. Principalele dezavantaje sunt costul și dificultatea manevrării și instalării. Acest mediu este folosit cu preponderență pentru legături punct la punct la distanțe mari (peste câteva sute de metri).

Material cu drept de autor

Un sistem de transmisie pe fibră optică este format dintr-un emiţător (LED sau laser), o fibră transportoare şi un receptor. Semnalul pe fibră optică este, de fapt, unda luminoasă emisă de un LED sau de un laser, în funcţie de tipul de fibră.

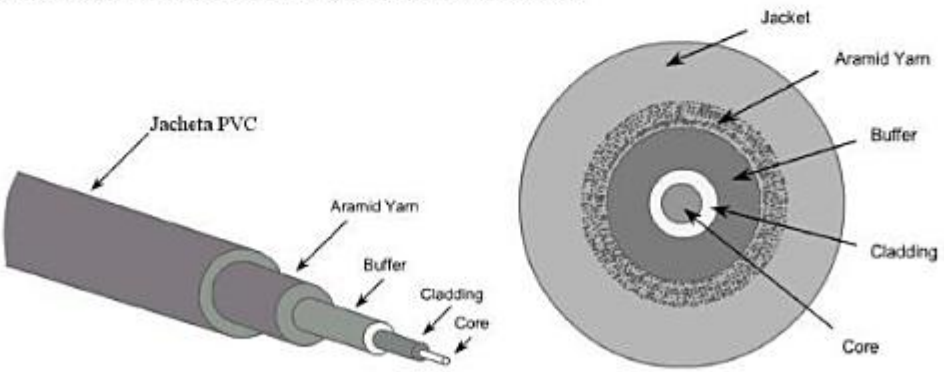
S-a observat că pentru anumite lungimi de undă semnalul suferă o atenuare mai mică decât pentru altele. În urma studiilor, s-au stabilit trei intervale („ferestre") pentru valorile lungimilor de undă la care atenuarea este foarte scăzută şi care permit emiţătorului să genereze mai multe semnale luminoase, iar receptorului să detecteze mai multe semnale. Aceste intervale sunt prezentate în graficul de mai jos.



1-12: Intervalele de lungimi de undă pentru care atenuarea este minimă

Notaţiile OH- indică faptul că la acele lungimi de undă în mod special, prezenţa ionilor OH- din materialul fibrei optice produc creşteri foarte mari ale atenuării. De aceea, lungimile de undă utilizate în sistemele optice sunt: 850nm, 1300nm (pentru fibra *multi-mode*) şi 1310nm, 1550nm (pentru *single-mode*).

Interiorul fibrei optice este format din miez (*core*) şi înveliş (*cladding*), două tuburi concentrice de sticlă, inseparabile, având indici de reflexie diferiţi. Propagarea semnalului se bazează pe fenomenul de reflexie totală. *Cladding*-ul, foarte subţire, cu diametrul de 125 microni, este învelit în trei straturi protectoare: un strat numit *buffer*, de obicei colorat, un înveliş rezistent de protecţie fabricat din kevlar (din acest material se fabrică şi vestele anti-glonţ) numit *Aramid Yarn* şi un înveliş exterior din PVC (*jacket*). Aceste trei straturi au rol de protecţie pentru partea din sticlă care este foarte fragilă.



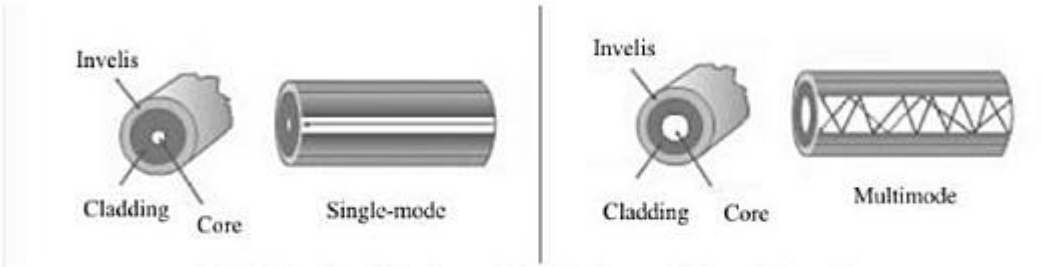
1-13: Structura fibrei optice

Material cu drept de autor

În funcţie de modul de transmisie şi, implicit, de dimensiunea *core*-ului, fibrele optice se împart în două categorii: *single-mode* şi *multi-mode*.

1.3.1 multi-mode

Fibra *multi-mode* are dimensiunea core-ului de 50 sau 62,5 microni, acest lucru permiţând transmiterea semnalului prin reflexie în pereţii core-ului. Acest tip de fibră permite distanţe mai mici decât cea *single-mode* (deoarece lumina are un drum mai lung de parcurs), însă este mai ieftină şi mai uşor de folosit (mai uşor de terminat cu conectori şi de sudat). De asemenea, echipamentele care emit semnal pe fibra optică *multi-mode* sunt mai ieftine, deoarece folosesc LED-uri (*light emitting diode*) cu lungimi de undă de 850 sau 1300 nanometri. Aceste echipamente cu LED-uri nu sunt periculoase pentru oameni (nu afectează ochii).



1-14: Structura fibrelor optice *single-mode* şi *multi-mode*

1.3.2 single-mode

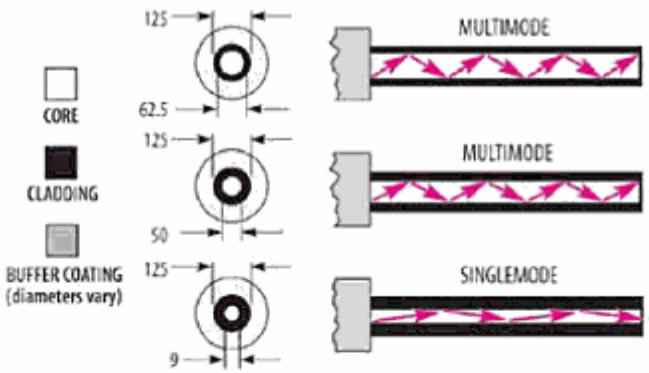
Fibra optică *single-mode* are o dimensiune a *core*-ului de 10 microni (mai nou între 5 şi 8 microni), acesta acţionând ca un ghidaj pentru raza luminoasă a semnalului care se transmite astfel aproape fără reflexie. Echipamentele terminale folosesc pentru a emite semnale luminoase lasere cu lungimi de undă de 1310 sau 1550 nanometri. Deoarece laserul emite o undă luminoasă foarte puternică şi focalizată, aceste echipamente pot produce leziuni grave ochiului. Aşadar, dacă vrem să vedem lumină într-o fibră optică, cel mai bine ar fi să alegem un echipament *multimode* sau o lanternă!

1.3.3 Comparaţie între single-mode şi multi-mode

Fibra optică *single-mode* permite distanţe mai mari de transmisie decât cea *multi-mode*, însă este mult mai scumpă şi impune precauţii speciale. De asemenea, echipamentele pentru *single-mode* sunt mai scumpe decât cele pentru *multi-mode*.

Din punct de vedere al vitezei maxime de transmisie, limita fizică este impusă de tehnologia folosită de echipamentele terminale, mai exact de viteza cu care sunt convertite impulsurile electrice în semnal optic, limita teoretică a lăţimii de bandă pe fibra optică în sine fiind foarte mare (~80 Tbps). Deşi, de exemplu, standardul *Ethernet* 802.3 pentru transmisie pe fibra optică limitează lungimea unui segment de fibră optică *multi-mode* la 2 km şi unul de *single-mode* la 3 km, trebuie menţionat că aceste limite se referă la modul de funcţionare CSMA-CD (atunci când sunt posibile coliziuni). Deoarece în cazul legăturilor de fibră optică sunt implicate conexiuni punct la punct, unde transmisia este *full-duplex* şi nu există posibilitatea apariţiei coliziunilor, limitarea distanţei maxime la care se poate întinde un segment de fibră optică este dată numai de puterea de emitere a dispozitivelor terminale, putând ajunge în cazul transmisiei *single-mode* şi la 120 de km pentru *FastEthernet* şi mult mai mult pentru alte tehnologii.

Material cu drept de autor



1-15: Categorii de fibră şi moduri de propagare

O comparaţie între laserele semiconductoare şi LED-uri ca surse de lumină este prezentată în tabelul de mai jos:

Criteriu	LED	Laser
Lungimea de undă folosită	850nm sau 1300nm	1310nm sau 1550nm
Tip de fibră	Multimode	Singlemode
Viteza de transfer a datelor	Mică	Mare
Distanţă	Scurtă	Lungă
Cost	Redus	Ridicat
Durată de viaţă	Lungă	Scurtă

1.3.4 Mod de construcţie, conectori

Procedeul industrial de construcţie al fibrei optice este foarte delicat şi, de aceea, foarte scump. Acest procedeu se numeşte OVD (*Outside Vapor Deposition*), iar fibra rezultată este sintetică şi are o consistenţă şi o geometrie extrem de precise. În linii mari, prin diferite procese chimice şi la temperaturi foarte înalte, se obţin doi cilindri concentrici de sticlă foarte pură, după care cilindrul astfel rezultat se trage şi se alungeşte până când se obţine o fibră care este rulată pe o rolă mare. Procesul este continuu, adică pe măsură ce se trage, se rulează fibra obţinută pe rolă. Acum se explică de ce fibra optică *single-mode* este mai scumpă decât cea *multi-mode*.

Fibra optică folosită în exteriorul clădirilor este diferită de fibra pentru cablarea de interior. Pentru cablările de exterior se foloseşte fibra *loose-tube* ce conţine mai multe perechi de fibre, fiecare dintre acestea având doar *core* şi *cladding*. O fibră de exterior poate conţine de la câteva perechi până la mii de perechi de fibre, costul cel mai mare pentru o instalare de exterior fiind manopera şi nu fibra propriu-zisă.

Pentru cablarea de interior putem întâlni două tipuri de fibră: cabluri cu mai multe perechi, numite *tight-buffer* şi cabluri cu o singură fibră, numite *patch*-uri. Cablurile cu mai multe perechi sunt folosite pentru cablarea orizontală, în vreme ce *patch*-urile sunt folosite pentru interconectarea dispozitivelor pe distanţe mici.

1.3.4.1 Îmbinări

Prin îmbinare (*splicing*) se înţelege conectarea permanentă a două cabluri de fibră optică. Îmbinările se realizează cu ajutorul unor dispozitive numite *splice*-uri. Caracterul permanent al

Material cu drept de autor

îmbinării este cel care face diferența între un conector și un *splice*. Totuși, terminologia poate crea confuzii, deoarece există producători ce oferă și *splice*-uri nepermanente, care pot fi decuplate în scopul efectuării unor reparații sau rearanjări.

Îmbinările sunt necesare, spre exemplu, pentru realizarea unor cabluri cu lungimi mai mari decât cele predefinite. Se poate întâmpla adesea ca un instalator de fibră optică să aibă în stoc mai multe cabluri cu diverse lungimi (în general producătorii oferă cabluri de lungime limitată – maxim 6 km) dar să nu aibă unul de 10 km. Cu ajutorul *splice*-urilor se pot îmbina două sau mai multe segmente pentru a obține cablul de lungimea dorită.

Realizarea îmbinărilor necesită o aliniere foarte precisă a celor două *core*-uri, astfel încât, la trecerea luminii prin punctul de joncțiune, să se piardă cât mai puțină energie. Cu alte cuvinte, trebuie ca aproape toată lumina venită pe o fibră să ajungă în *core*-ul celei de-a doua. Contactul efectiv între cele două tuburi de sticlă nu este neapărat necesar. Cea mai mare provocare pentru *designer*-ii de *splice*-uri este dată de necesitatea alinierii foarte precise.

Există două mari tipuri de îmbinări: mecanice sau prin sudură.

Îmbinările prin sudură folosesc un arc electric pentru a topi și suda cele două fibre de sticlă. Aceste suduri implică o procedură complicată de aliniere, controlată prin calculator, și reușesc să limiteze pierderile la doar 0,05dB. Costurile manoperei pentru acest tip de îmbinare sunt, însă, foarte ridicate, la fel și costurile de timp.

Îmbinările mecanice sunt rapid de implementat și nu necesită o instruire prealabilă, însă pierderile sunt de aproape 0,2dB.

1.3.4.2 Conectori

Un cablu de fibră optică poate fi terminat în două feluri – folosind *splice*-uri prin care se realizează îmbinări permanente sau nepermanente între două fibre sau folosind conectori pentru cuplarea cablului la un echipament de rețea. Aceste terminații trebuie să fie alese în conformitate cu tipul fibrei și instalate astfel încât să minimizeze pierderile de lumină și să nu permită pătrunderea impurităților. Întrucât fibra optică a apărut la sfârșitul anilor `70, producătorii au scos pe piață peste 80 de modele de conectori și numeroase metode de instalare, fiecare încercând să scadă cât mai mult atenuarea (pierderea de semnal) și reflexia (apariția de semnale reziduale). Dintre acestea, însă, doar câteva tipuri sunt folosite în mod curent. Cei mai populari conectori sunt cei de tip ST (*Straight Tip*) și SC (*Subscriber Connector*).

Conectorul de tip ST, apărut în 2005, are o formă circulară, asemănătoare într-o anumită măsură cu BNC-ul și este încă folosit pentru rețelele multimode. Cilindrul (ferula) care susține fibra are 2,5 mm, la fel ca majoritatea conectorilor și este confecționat cel mai adesea din ceramică sau metal și rareori din plastic. Întrucât îmbinarea se face prin presare, se poate întâmpla să nu fie poziționat corect și, de aceea, în caz că se sesizează pierderi prea mari, trebuie scos și reconectat. Din păcate, acest conector ocupă loc mult și, de aceea, conectorul recomandat în acest moment este SC, care are o formă dreptunghiulară și o conectare de tip *push-pull*.

SC a fost definit în standardul TIA-568-A, dar nu a fost folosit la început deoarece costa de două ori mai mult decât un conector ST. În ziua de azi este cel mai popular datorită performanțelor sale foarte bune, a manipulării foarte facile și a prețului aproape egal cu cel al unui conector ST. Este disponibil și în varianta pentru configurații duplex. Trebuie menționat că transmisia pe fibră optică se face pe o pereche (un fir pentru TX și unul pentru RX); conectorii duplex permit terminarea ambelor fibre în aceeași mufă.

Conectorii impun o atenție sporită la terminarea cablurilor de fibră optică, deoarece punctele de joncțiune sunt cele care introduc cea mai mare atenuare și unde se poate întâmpla ca lumina fi reflectată înapoi în fibră.

Material cu drept de autor

1-16: Conectori ST (sus) şi SC (jos)

Pentru a face conectorii mai uşor de recunoscut, standardul TIA-568 specifică un cod al culorilor în care conectorii pentru fibră *multi-mode* au culoarea bej, iar cei pentru *single-mode* sunt albaştri.

1-17: Conectori ST şi SC

1.3.4.3 Analiza performanţelor unei legături pe fibră optică

Există mai multe metode pentru calculul atenuării şi pentru estimarea distanţei maxime în cazul unei legături prin fibră optică. Cea mai simplă şi mai precisă metodă este folosirea unui *Optical Time Domain Reflectometer* (OTDR). Cu ajutorul acestui instrument se obţine o valoare exactă pentru întreaga energie ce se pierde prin atenuare (atât atenuarea mediului cât şi cea introdusă de conectori sau de *splice*-uri). În lipsa unei caracterizări riguroase date de un OTDR, atenuarea unei legături poate fi estimată dacă sunt cunoscute lungimea fibrei şi variabilele de atenuare.

Variabilele de atenuare sunt conectorii, *splice*-urile şi rata de atenuare pe kilometru specifică fibrei. Dacă nu pot fi cunoscute valorile exacte pentru toate variabilele, este necesară o estimare a acestora, şi anume, luarea în calcul a cazului cel mai nefavorabil. Tabelul de mai jos include valorile atenuării stabilite prin convenţie (EIA/TIA) pentru variabilele de atenuare.

Atenuarea introdusă de un conector se estimează la 0,75dB, cea introdusa de un *splice* mecanic 0,2dB, iar cea apărută în cazul unei suduri 0,05dB.

Valorile aproximative de mai sus repretintă cazul cel mai defavorabil. Fiecare producător de echipamente pentru fibră optică încearcă să reducă cu cât mai mult atenuarea pentru fiecare dintre variabile.

Material cu drept de autor

Tipul de fibră	Lungimea de undă	Atenuarea / km
Multimode 50/125µm	850 nm	3,5 dB
	1300 nm	1,5 dB
Multimode 65,5/125 µm	850 nm	3,5dB
	1300 nm	1,5 dB
SingleMode 9µm	1310nm	0,4dB
	1550nm	0,3dB

1-18: Categorii de fibră optică

Pentru a calcula atenuarea totală se înmulțește lungimea cablului (în km) cu atenuarea corespunzătoare tipului de fibră și se adună atenuarea introdusă de fiecare *splice* sau conector de pe legătură. De exemplu, pentru un cablu de 40 de km de fibră *single-mode* la 1310nm cu 2 conectori și 5 *splice*-uri de tip sudură, atenuarea totală se calculează: 40km x 0,4dB/km + 0,05dB x 5 + 0,75dB x 2 = 17,75dBm.

La această valoare se recomandă adăugarea unei marje de siguranță de cel puțin 10dB deoarece se poate întâmpla ca estimările să fi fost prea optimiste, specificațiile *vendor*-ului inexacte sau, pur și simplu, atenuarea introdusă de anumite componente să nu fi fost luată în calcul. Ceea ce înseamnă că este nevoie de o putere de aproximativ 27,75 dBm pentru ca semnalul să ajungă la destinație peste nivelul minim de sensibilitate al receptorului. „dBm" este unitatea folosită în exprimarea puterii măsurate raportată la un miliWatt.

Bugetul optic reprezintă diferența dintre puterea minimă de transmisie a emițătorului și sensibilitatea receptorului. Este important ca după ce legătura a fost stabilită, să se măsoare și să se verifice valorile efective ale atenuării, pentru a identifica potențialele probleme de performanță.

Distanțele recomandate de către IEEE pentru un cablu de fibră optică în funcție de standardul Ethernet care va fi folosit sunt prezentate în tabelul următor.

Standard	Bandwidth	Λ	Tipul de fibră	Distanța recomandată
10BASE-FL	10	850nm	Multimode 50/125 µm sau 65,5/125 µm	2 km
100BASE-FX	100	1300nm	Multimode 50/125 µm sau 65,5/125 µm	2 km
100BASE-SX	100	850nm	Multimode 50/125 µm sau 65,5/125 µm	300 m
1000BASE-SX	1000	850nm	Multimode 50/125 µm	550 m
			Multimode 65,5/125 µm	220 m
1000BASE-LX	1000	1300nm	Multimode 50/125 µm sau 65,5/125 µm	550 m
		1310nm	Singlemode 9/125 µm	5 km
1000BASE-LH	1000	1550nm	Singlemode 9/125 µm	70 km

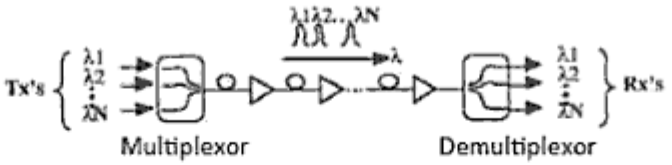
1.3.5 Multiplexarea prin divizarea lungimii de undă – WDM

În secțiunile anterioare s-a discutat despre multiplexare, procedeul prin care mai multe canale de date sunt combinate într-un singur canal.

WDM (*Wavelength Division Multiplexing*) este o formă de multiplexare pentru canalele de fibră optică ce duce la o creștere semnificativă a capacității de transmitere a datelor în mediul optic. Principiul de funcționare al acestei tehnologii îl reprezintă multiplexarea mai multor raze optice într-un singur canal (fibră comună) printr-un sistem complex de oglinzi și folosirea unor lungimi de undă diferite. Cât timp fiecare canal are propriul domeniu de frecvență (lungime de

Material cu drept de autor

undă) şi toate aceste domenii sunt disjuncte, ele pot fi multiplexate împreună pe o fibră pe distanţă foarte mare.



1-19: WDM

Sistemul WDM foloseşte pentru transmisia pe distanţe foarte mari un multiplexor la transmiţător pentru combinarea semnalelor pe un canal comun şi un demultiplexor la receptor pentru a le despărţi. Fiecare fibră de la destinaţie conţine un filtru special (construit folosind o prismă), care filtrează toate lungimile de undă mai puţin una. Semnalele rezultate pot fi rutate către destinaţie sau recombinate în diferite feluri pentru transmisii ulterioare.

Concept publicat încă din anii 1970, tehnologia WDM a progresat extrem de rapid. Dacă primul sistem WDM combina doar 2 canale, sistemele moderne pot combina până la 160 de semnale, putând astfel extinde un sistem de 10Gbps până la o valoare teoretică de 1Tbps doar pe o pereche de fibră optică. În 2001 existau produse pe piaţă cu 96 de canale de 10Gbps fiecare, deci un total de 960Gbps. În laboratoare se lucrează deja la sisteme ample ce cuprind peste 200 de canale. Atunci când numărul de canale este foarte mare şi lungimile de undă sunt foarte apropiate (0,1nm) sistemul este numit DWDM (*Dense WDM*).

1.3.6 Comparaţie între fibra optică şi cablul UTP

La începuturile fibrei optice au existat păreri conform cărora în „câţiva" ani firele de cupru vor fi înlocuite în totalitate cu fibră optică. Acest lucru s-a dovedit greşit. Printre avantajele pe care le prezintă firele de cupru se numără: preţul scăzut, instalarea facilă, faptul că nu necesită atenţie sporită în utilizare. Aceste avantaje fac firele de cupru mediul ideal pentru cablări în reţele mici şi mijlocii, în interiorul clădirilor, unde nu se justifică fibra optică. Printre dezavantajele majore ale firelor de cupru se numără: sunt susceptibile la interferenţe electrice şi pot fi folosite pe distanţe relativ mici - oricum mult, mult mai mici decât echivalentul lor în fibră optică.

Fibra are multe avantaje. În primul rând, lărgimea de bandă pe care o suportă este mai mare decât a cuprului. Un singur cablu de fibră optică *multi-mode* (ce conţine mai multe fibre) poate purta acum aproape 5 milioane de convorbiri telefonice simultane. Fibra are avantajul că nu este afectată de şocurile electrice, de interferenţa câmpului electromagnetic sau de căderile de tensiune. De asemenea, nu este afectată de substanţele chimice corozive din aer, fiind ideală pentru mediile aspre din fabrici. Companiile de telefoane preferă fibra şi din alt motiv: este subţire şi foarte uşoară. Canalele cu cabluri sunt, în general, pline până la refuz; prin înlocuirea cuprului cu fibră se golesc canalele, iar cuprul are o valoare foarte bună pe piaţă. În plus, 900 de cabluri torsadate de 1 km lungime cântăresc 7250 kg. Un cablu ce conţine 24 fibre şi are aceeaşi capacitate cântăreşte doar 60 kg, acest lucru reducând drastic necesitatea unor echipamente mecanice scumpe care trebule întreţinute. În fine, fibrele optice introduc o atenuare neglijabilă şi sunt foarte dificil de interceptat. Acest lucru le oferă o excelentă securitate. Motivul pentru care fibra este mai bună decât cuprul este intrinsec. Electronii în mişcare dintr-un cablu interacţionează cu alţi electroni şi sunt influenţaţi de alţi

Material cu drept de autor

electroni din afara cablului. Fotonii dintr-o fibră nu interacționează între ei și nu sunt afectați de fotonii din exterior.

Pe de altă parte, fibra este o tehnologie mai puțin familiară și necesită o pregătire pe care mulți ingineri nu o au. Terminarea fibrei (adică atașarea conectorilor) este un procedeu dificil care necesită multă pregătire și experiență. De asemenea, fibra optică este suficient de pretențioasă și, de aceea, necesită o utilizare mai atentă decât cablul UTP (nu trebuie îndoită prea tare, călcată sau strânsă după piciorul mesei, etc). Deoarece transmisia optică este prin natura ei unidirecțională, comunicațiile bidirecționale necesită fie două fibre, fie două benzi de frecvență diferite pe aceeași fibră. Nu în ultimul rând, interfețele pentru fibră costă mult mai mult decât interfețele electrice.

Cu toate acestea, este foarte probabil că în viitor toate comunicațiile de date pe lungimi mai mari de câteva zeci de kilometri se vor face prin fibră optică.

1.4 Caracteristici ale mediilor de transmisie

1.4.1 Frecvența

Frecvența este, probabil, cel mai important parametru al mediului de transmisie. Ea este cea care arată câte semnale pot fi puse pe mediu în unitatea de timp, așadar, care este cantitatea maximă de informație ce ar putea fi transferată.

1.4.2 Lățimea de bandă

Termenul de lățime de bandă (*bandwidth*) poate fi interpretat în două feluri. Unul dintre sensuri este acela al diferenței dintre două niveluri de frecvență, adică dimensiunea unei benzi de frecvență (măsurată în Hertzi).

Cel de-al doilea sens al termenului indică numărul maxim de biți transferați sau procesați în unitatea de timp. Tabelul de mai jos poate da o idee despre ce lățime de bandă ocupă diverse servicii oferite de rețelele actuale.

Voce (VoIP)	0,1 Mbps
Navigare pe internet	5 Mbps
Interacțiuni video, jocuri, EoD (Entertaiment on Demand) , video conferințe	10 Mbps
HDTV, IPTV, VoD (Voice on Demand) 1 - 3 canale	30 Mbps

Din acest tabel putem observa că în viitor, pentru a beneficia și de servicii HDTV, cerințele minime ale unei rețele în ceea ce privește lățimea de bandă vor depăși 50 Mbps.

1.4.3 Unități de măsură

Unitățile de măsură reprezintă una dintre cele mai frecvente surse de erori în discuțiile despre rețelele de calculatoare. Erorile țin atât de unitățile de măsură propriu-zise, cât și de multiplii acestora.

Prima confuzie apare în exprimarea vitezei de transfer. Capacitatea mediului de transmisie este măsurată în **biți pe secundă**, în vreme ce cantitatea de date transferată este cel mai adesea exprimată în **octeți pe secundă**. În plus, notațiile celor două unități de măsură sunt foarte similare: în primul caz se folosește notația bps, iar pentru transferul de date Bps.

Atunci când un modem se conectează la 33,6 k, aceștia sunt kbps, iar prin împărțirea la 8 se obțin 4,2 kiloBytes. Pentru acest caz, dacă fereastra *browser*-ului web arată o viteză de

Material cu drept de autor

descărcare de 4 k, adică 4 kBytes, conexiunea este considerată de bună calitate. Cu toate acestea, se poate întâmpla ca viteza de transfer afișată să fie mai mare decât viteza modemului. Explicația cea mai probabilă a unei astfel de situații este că datele transferate sunt necomprimate (fișiere text, spre exemplu, precum codul paginilor HTML) și modemul sau aplicația de transfer realizează o compresie a acestora.

Un alt factor în diferența dintre viteza mediului și viteza de transfer a datelor este cantitatea de informație adăugată de stiva de protocoale prin diferitele antete. Pentru o transmisie ce folosește TCP/IP peste *Ethernet* această informație suplimentară variază între 58 și 98 de octeți, ceea ce pentru cadru de maxim 1500 de octeți specificat de *Ethernet* înseamnă că până la 6,5% din informația transferată nu e informație utilă. Această valoare este și mai mare dacă nu se folosesc doar pachete de dimensiune maximă. Altfel spus, pentru o rețea *Ethernet*, deși aceasta dispune de o lățime de bandă de 10 Mbps, viteza de transfer a datelor este de aproximativ 1,15 MBps.

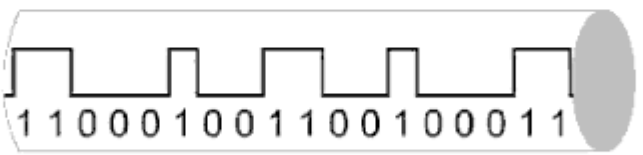
O a doua confuzie importantă apare în exprimarea multiplilor unităților de măsură. În transmisia de date un kilobit reprezintă 1000 de biți, în vreme ce din punctul de vedere al sistemelor de operare un kilobit este compus din 1024 de biți.

Noțiunea de **baud** era folosită ca unitate de măsură în exprimarea vitezei pentru transmisiile de date, în special pe legăturile seriale. Numărul de bauzi indica numărul de schimbări pe secundă ale stării mediului de transmisie (de schimbări ale nivelului de tensiune). Întrucât o astfel de schimbare sesizată pe mediu poate fi interpretată nu doar ca un singur bit, ci și ca doi sau mai mulți, în funcție de modularea folosită, unitatea de măsură adecvată este cea de bps (biți pe secundă).

1.4.4 Baseband și broadband

Termenii de *„baseband"* (în bandă de bază) și *„broadband"* (în bandă largă) descriu numărul de canale de comunicație folosite pe un anumit mediu de transmisie. Cu alte cuvinte, pe un fir de cupru transmisia poate fi făcută pe un singur canal de comunicație (cazul *baseband*) sau pe mai multe canale (cazul *broadband*).

În cazul comunicației în **bandă de bază**, pe mediul de transmisie există un sigur semnal. Acel semnal poate avea mai multe componente, însă din punct de vedere al firului de cupru sau al fibrei optice este un singur semnal (electric sau optic).

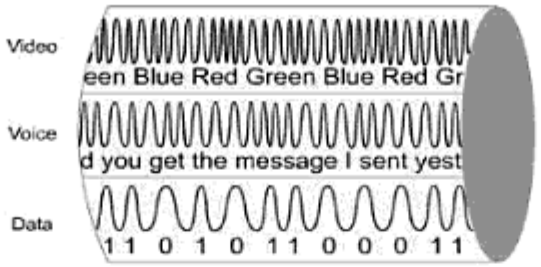


1-20: Transmisie *baseband* (în banda de bază)

De exemplu, în telefonia fixă din România mediul fizic de transmisie este alcătuit din două fire torsadate. Semnalul este vocea umană transmisă prin intermediul telefonului; în cazul în care este folosit un modem pentru *dial-up*, semnalul constă din datele transmise de calculator. Acest sistem de comunicație este de tip *baseband,* deoarece cele două comunicații nu pot avea loc simultan. Dacă însă se utilizează o conexiune DSL, atunci pe același mediu fizic pot fi realizate simultan și telefonie și transmisie de date.

În cazul comunicației în **bandă largă**, pe același mediu fizic există mai multe canale de comunicație independente, multiplexate într-un singur semnal *broadband*.

Material cu drept de autor



1-21: Transmisie *broadband* (în bandă largă)

Un exemplu de transmisie *broadband* foarte des întâlnită este CATV. CATV (*community antenna television*) este, de fapt, sistemul de televiziune prin cablu comun. Principiul de funcţionare este simplu: multiplexarea în frecvenţă. Fiecare canal de televiziune are alocată o bandă de frecvenţă de 6 MHz. Firma de televiziune prin cablu recepţionează practic posturile TV de la diferite surse, prin diferite metode (cablu de cupru, fibră optică, antene radio) şi compune aceste semnale independente într-un singur semnal *broadband*, folosind o tehnică optimizată de multiplexare în frecvenţă. Acest semnal *broadband* este trimis pe cablul coaxial ce ajunge acasă, în televizor. Acesta, la rândul sau, utilizează un demultiplexor ce separă semnalul *broadband* primit în canalele independente iniţiale.

Caracteristicile conexiunilor *broadband* se redefinesc în permanenţă, odată cu trecerea timpului, şi variază în funcţie de ţară. În 2004, spre exemplu, o conexiune *broadband* în Anglia trebuie să ofere minim 2 Mbps, în Germania şi Statele Unite serviciile *boadband* încep de la 4 Mbps, în vreme ce în Japonia majoritatea furnizorilor de servicii *broadband* oferă minim 20 Mbps

1.5 Echipamente de reţea de nivel fizic

Deoarece la nivel fizic nu există date, ci doar semnale, aceste echipamente nu fac decât să prelucreze semnalul fizic, fără să încerce să interpreteze datele transmise prin acel semnal.

1.5.1 Repetorul

Principala funcţie a repetorului este aceea de a extinde suprafaţa acoperită de o reţea. Repetorul este un echipament care primeşte un semnal şi îl retransmite la o putere mai mare, împiedicând ca atenuarea sa atingă o valoare prea mare, sau îl redirecţionează, pentru ca semnalul să poată ocoli un obstacol. Sunt importante aspectele legate de cost şi, în special, de latenţa introdusă. Ambele trebuie să fie cât mai mici.

Un repetor digital amplifică, restaurează, sincronizează sau aplică orice combinaţie a acestor funcţii asupra unui semnal digital.

Comunicaţiile intercontinentale sau cele pe sub ocean ar fi imposibile fără existenţa repetoarelor.

1.5.1.1 Pentru cupru - Hub

Hubul sau repetorul multiport poate conecta mai multe cabluri, astfel încât toate vor face parte dintr-un segment de reţea partajat. Întrucât hubul este un dispozitiv de nivel fizic, el nu interpretează în niciun fel semnalul, ci doar îl trimite mai departe pe toate porturile, cu excepţia celui pe care a fost recepţionat. În mod evident, aceasta duce la un risc foarte ridicat al producerii de coliziuni, tratarea lor rămânând devenind responsabilitatea fiecărei staţii conectate.

Material cu drept de autor

Odată cu scăderea prețurilor la switchuri, huburile au început să nu mai fie folosite în rețele locale; ele mai pot fi văzute doar în infrastructuri mai vechi.

1.5.1.2 Repetoare optice

După cum s-a observat, limitările distanței maxime la care se poate întinde un segment de fibră optică sunt date numai de puterea de emitere a dispozitivelor terminale. De aceea, pentru extinderea ariei de acoperire în sistemele de fibră optică se folosesc repetoare și amplificatoare optice, capabile să regenereze semnalele pe distanțe de sute de kilometri.

Un repetor folosit în comunicația prin fibră optică, cunoscut și sub numele de OEO (*Optical-Electrical-Optical*) este un dispozitiv a cărui principală funcție este să primească semnalul optic, să îl transforme în semnal electric, să îl prelucreze și apoi să emită din nou semnalul optic, eliminând astfel atenuarea acestuia ce poate duce la erori de interpretare în cazul transmisiilor pe distanțe foarte mari. În ziua de azi, din cauza eficienței scăzute și a costului mare de implementare, repetoarele au fost înlocuite cu amplificatoare optice (EDFA - *Erbium Doped Fiber Amplifier*). Acestea pot amplifica semnalul o dată la 1000 km fără a mai fi nevoie de conversii multiple între semnalele electrice și cele optice. Un amplificator optic crește puterea semnalului fără să îl transforme în semnal electric, ceea ce înseamnă că nu îl regenerează, ci doar îl amplifică. Această tehnică este posibilă deoarece în mediul optic atenuarea este cea care limitează distanțele și nu distorsionarea semnalului.

Inovațiile recente în domeniul fibrei optice și al comunicațiilor prin fibră optică au redus degradarea semnalului optic atât de mult încât nevoia de regenerare a acestuia apare pe distanțe ce depășesc câteva sute de kilometri. Aceasta a crescut eficiența rețelelor optice, mai ales a celor ce se întind pe sub oceane (Atlantis-2, TGN Atlantic, Hibernia), unde costul și eficiența amplificatoarelor este unul dintre factorii cheie ce determină performanța întregului sistem de cablare[1].

1.5.1.3 Repetoare *wireless*

Pe lângă regenerarea semnalului, repetoarele *wireless* sunt folosite și pentru ghidarea acestuia. Spre exemplu, dacă între două centre de emisie radio există un obstacol (un deal mai înalt, sau un munte) și ele nu se „văd", comunicația nu este posibilă. Pentru ca semnalul radio să poată fi recepționat și de cealaltă parte a obstacolului, este necesară instalarea, chiar în vârf, a unui repetor *wireless* care să preia semnalul de pe un versant și să îl emită către celălalt. Această tehnică este folosită pentru orice fel de transmisii fără fir: radio, televiziune, telefonie mobilă, etc. Frecvențele undelor cu care se operează sunt cuprinse între frecvențele undelor radio și ale microundelor.

Din punct de vedere al alimentării cu energie electrică, repetoarele *wireless* se împart în active și pasive. Cele active constau în antene direcționale instalate în locații cu înălțimi cât mai mari (turnurile radio, de televiziune, etc), ce permit accesul la o sursă de alimentare electrică. În locurile greu accesibile (vârfuri de munte, deșert, etc) sunt montate repetoarele pasive, care nu necesită o sursă de energie electrică, însă propagă un semnal mult mai slab.

Există și repetoare radio pentru radio-amatori sau pentru cei care vor să extindă zona de acoperire a echipamentelor *wireless* de putere mică. Acestea acoperă distanțe mai scurte (de ordinul sutelor de metri).

[1] http://www.atlantic-cable.com/Maps/index.htm

Material cu drept de autor

1.5.2 Convertorul

Convertorul sau *transceiver*-ul - termen provenind din combinarea lui *trans*(mitter) cu (re)*ceiver* - oferă posibilitatea inteconectării a două medii de comunicație diferite. Convertoarele sunt clasificate în două categorii: convertoare pasive și convertoare active, cele din urmă necesitând alimentarea la o sursă de curent pentru a putea funcționa.

1.5.2.1 Convertoare pasive

În categoria convertoarelor pasive sunt convertoarele ce fac trecerea de la conectorul DB-15 (AUI), la UTP sau BNC. Deși rețelele locale pe cablu gros au dispărut odată cu venirea anilor '90, interfețele AUI au fost interfețele *Ethernet* standard pentru routerele oferite de majoritatea producătorilor de echipamente de rețea, până spre sfârșitul anilor '90. Pentru folosirea unei astfel de interfețe într-o rețea locală este nevoie de un convertor. Din păcate, prețul unui convertor AUI a rămas constant de aproape 10 ani, în jurul valorii de 25 de USD.

Convertoarele UTP – BNC nu s-au bucurat de o popularitate prea mare datorită integrării funcției de conversie între mediul coaxial și cel torsadat la nivelul hubului, precum și a tendinței continue de scădere a prețurilor pentru huburi.

1.5.2.2 Convertoare electric – optic

Cel mai frecvent întâlnite convertoare active fac trecerea de la mediul optic la cel electric. Aceste convertoare se numesc MC (*Media Convertor*), deși uneori sunt numite tot *transceivere*. O alternativă la folosirea unui astfel de convertor o reprezintă switchurile, ce oferă atât interfețe optice, cât și porturi RJ-45. Totuși, costul unui astfel de switch depășește 1000 USD, în vreme ce costul unui MAU (*Medium Attachement Unit*) oscilează în jurul a 100 USD.

1.5.2.3 Convertoare electric – *wireless*

Dispozitivul care transformă semnalul electric primit pe fir de cupru în undă electromagnetică de înaltă frecvență pe care o împrăștie în aer este *Acces-Point*-ul. Deoarece standardele de transmisie *wireless* conțin specificațiile de nivel fizic (distanța, lățimea de bandă, puterea de transmisie) în strânsă legătură cu cele de nivel legătură de date, s-a optat pentru prezentarea soluțiilor de comunicație fără fir într-un capitol dedicat.

1.6 Studii de caz

1.6.1 Realizarea *patch*-urilor UTP *straight-through, crossover, rollover*

Pentru sertizarea unui cablu UTP CAT5 este necesar un clește de sertizat și de un conector 8P8C (numit și RJ45). Prima operație constă în înlăturarea izolației din jurul firelor din cablu. Trebuie acordată o atenție deosebită la detorsadarea firelor: atunci când se îndepărtează manșonul de plastic și se detorsadează perechile pentru a putea introduce firele în mufă, trebuie ca bucata de cablu detorsadat sa fie cât mai mică. În caz contrar, va apărea o interferență între fire ce produce efectul de *crosstalk*. Practic, se taie 3-4 cm din manșon, se detorsadează firele, se aranjează în ordinea dorită, iar apoi, cu ajutorul unor lame ale cleștelui de sertizat, se taie firele astfel încât dimensiunea zonei neizolate să reprezinte aproximativ ¾ din lungimea mufei. În acest fel, firele vor ajunge până în capătul mufei asigurând un contact electric perfect, iar bucata detorsadată va fi aproape inexistentă, minimizând riscul apariției *crosstalk*-ului.

Mufele RJ-45, folosite pentru terminarea cablurilor UTP, conțin 8 lăcașuri în care trebuie aduse cele 8 fire. Pinii conectorului sunt niște lamele metalice care, inițial, se află deasupra

Material cu drept de autor

lăcașului, pentru ca firul de cupru să poată intra. Prin folosirea cleștelui de sertizat lamelele sunt împinse în lăcașurile unde se găsesc firele. Prin apăsare, lamelele vor străpunge firul de cupru, realizându-se astfel contactul electric.

Pentru realizarea unui *patch* UTP *straight-through* firele trebuie să se găsească în ambii conectori, fie în ordinea impusă de T568A, fie T568B. Același standard trebuie respectat și la un capăt și la celălalt.

Pentru realizarea unui *patch* UTP *crossover* perechea verde dintr-un capăt este inversată cu perechea portocalie din celălalt. Cu alte cuvinte, una din terminații respectă T568A, cealaltă T568B.

Pentru realizarea unui *patch* UTP *rollover*, se sertizează un capăt al cablului folosind unul dintre standarde, iar la celălalt capăt firele se așează în ordine inversă (în oglindă) – pinul 1 va corespunde pinului 8, pinul 2 va corespunde pinului 6, etc.

Ordinea firelor pentru cele două standarde se regăsește în figura 1-11.

Dacă nu se respectă standardul, există un risc major ca cele două fire folosite pentru Rx sau Tx să nu facă parte din aceeași pereche, și să nu-și mai anuleze reciproc câmpurile electrice. Practic, torsadarea nu mai acționează corect si sunt generate interferențe ce alterează semnalul electric. (Cu alte cuvinte ori nu va merge, ori va merge extrem de prost!)

În general, în Europa se folosește standardul 568B , iar în Statele Unite 568A. De ce este important de știut și de respectat acest lucru? Teoretic, nu contează care din acest standard este utilizat atât timp cât ambele mufe (de la cele două capete) sunt făcute folosind același standard. Practic însă, la construirea și administrarea unei rețele de mari dimensiuni lucrează mulți oameni, care nu întotdeauna comunică între ei. Așadar, pentru reducerea erorilor umane este necesară respectarea aceluiași standard.

Material cu drept de autor

2 Reţele Ethernet

Ce se învaţă din acest capitol?

- Ce reprezintă standardul 802.3
- Cum funcţionează CSMA/CD
- Modul de funcţionare al unui switch *Ethernet*
- Cum sunt evitate buclele de nivel 2 folosind STP
- Reţea locală virtuală (VLAN)
- Standardul 802.1Q

Cine este...

Norman Abramson *este vicepreşedintele ALOHAnet, prima reţea de calculatoare, înfiinţată la Universitatea din Hawaii. Ideea ALOHAnet este de a folosi o infrastructură de cost scăzut de tipul radioului pentru amatori pentru a lega calculatoarele universităţii, aflate la mare distanţă. Desi reţeaua nu se mai foloseste, a reprezentat un punct de pornire al Interetului. Norman Abramson a primit de la IEEE medalia Alexander Graham Bell.*

Bob Metcalfe *este co-inventator al protocolului Ethernet, fondator al 3Com şi al legii care îi poartă numele. În timp ce îşi susţinea doctoratul la MIT s-a implicat în conectarea universităţii la noua reţea ARPAnet, fiind responsabil cu găsirea unui hardware care să facă legatura. În timp ce a fost angajat la Xerox PARC a co-inventat Ethernetul. Dupa ce a plecat de la Xerox a pornit 3Com, companie ce fabrică componente de reţelistică. Legea lui Metcalfe afirmă că valoarea unei reţele de telecomnunicaţii este direct proporţională cu pătratul numărului de utilizatori.*

2.1 Noţiuni generale

Ethernet este în ziua de azi tehnologia dominantă de LAN, ce defineşte un număr de standarde pentru nivelul fizic, o metodă de acces la mediu (CSMA/CD) şi o schemă de adresare. Primul standard *Ethernet* a fost publicat în 1980 de un consorţiu format din firmele DEC, Intel şi Xerox, consorţiu numit DIX. Cu câteva modificări minore, acesta a devenit trei ani mai târziu standardul IEEE 802.3. Alte standarde IEEE importante sunt : 802.11 – *Wireless LAN*; 802.1d – definire cadre speciale (BPDU) şi funcţionarea acestora; 802.1q – standard pentru VLAN-uri ; 802.1x – standard de autentificare.

Pentru comunicarea în cadrul unei reţele *Ethernet*, ca şi în cazul altor standarde IEEE 802, fiecărei staţii i se atribuie o adresă MAC (*Media Access Control*) unică pe 48 de biţi exprimaţi în 12 cifre hexazecimale, ce este folosită pentru a specifica atât sursa cât şi destinaţia fiecărui pachet de date.

Adresa MAC este un şir de 48biţi folosit pentru asigurarea unicităţii în reţelele Ethernet.

O adresă MAC este stocată în memoria ROM şi este încărcată în RAM în momentul iniţializării plăcii de reţea. Din această cauză adresele MAC mai sunt numite şi adrese fizice sau *burned-in addresses* (BIAs).

Adresele MAC sunt adrese ce folosesc o schemă de adresare plată (spaţiul de adresare ocupat treptat şi complet). Cum stau lucrurile în realitate? Instituţia ce administrează adresele fizice este IEEE. Problema este că IEEE nu poate monitoriza direct atribuirea fiecărei adrese fizice, astfel încât transferă această responsabilitate producătorilor. Din cei 6 octeţi ce compun adresa fizică primii trei vor fi folosiţi pentru identificarea fabricantului, acest câmp fiind

Material cu drept de autor

denumit *Organizational Unique Identifier* (OUI), iar următorii trei octeți sunt dați de către fabricant în mod unic fiecărui dispozitiv de rețea. Astfel, spațiul adreselor fizice poate fi ordonat fără îndoială după producător, informație totuși inutilă, deoarece rar se întâmplă ca într-o rețea să existe dispozitive de rețea produse de un singur producător.

În concluzie mulțimea adreselor fizice este o mulțime neordonată, care în plus nu poate folosi integral spațiul de adrese. Cu toate acestea schema de adresare fizică este unul dintre puținele lucruri ce nu a trebuit schimbat și nici măcar actualizat pe parcursul ultimilor douăzeci de ani. IEEE a anunțat faptul că spațiul adreselor fizice nu va fi epuizat mai devreme de anul 2100[1]. Deși diferența între schema de adresare fizică (MAC - 48 de biți) și schema de adresare de nivel trei dominantă astăzi în Internet (IPv4), ce asigură identificarea fiecărei stații în mod unic printr-o adresă logică pe 32 de biți, nu este foarte mare (doar 16 biți), în ziua de astăzi există o nevoie de extindere a celei din urmă la 128 de biți (IPv6). Apariția și dezvoltarea rapidă a rețelelor de calculatoare personale va duce treptat la epuizarea adreselor IPv4, estimată de unii vizionari între anii 2019 – 2040.

Adresele fizice oferă suport pentru 3 tipuri de comunicație: directă (*unicast*), prin difuzare (*broadcast*) și cu destinație multiplă (*multicast*), primele două tipuri fiind mult mai populare decât ultimul tip de comunicație. Adresa de difuzare pentru nivelul legătură de date are o valoare unică, aceasta fiind: FF:FF:FF:FF:FF:FF.

Având numeroase avantaje (ușurința de instalare și întreținere, capacitatea de a introduce noi tehnologii, fiabilitatea, costul relativ scăzut), rețelele *Ethernet* sunt de tip *shared-media* (mediu multiacces – mai multe stații conectate la acelasi mediu fizic), deci orice cadru transmis de către o stație va fi recepționat de către toate celelalte stații din rețeaua locală.

2.1.1 Structura cadrului Ethernet

Din punctul de vedere al cadrului *Ethernet*, protocolul oferă trei tipuri de informații: identificarea destinației și a sursei pe baza unei adrese MAC, precizarea protocolului de nivel superior (de nivel rețea) și o sumă de control pentru verificarea integrității datelor.

Structura cadrului *Ethernet* este aproape identică, indiferent de varianta de *Ethernet* folosită, și conține următoarele câmpuri:

6	6	2	46-1500	4
Adresă destinație	Adresă sursă	Lungime/Tip	Date	Sumă de control

2-1: Structura cadrului *Ethernet*

Antetul *Ethernet* este de 14 octeți, împărțiți în trei câmpuri: 6 octeți pentru adresa destinație, 6 octeți pentru adresa sursă și 2 octeți pentru câmpul tip, câmp ce este folosit pentru precizarea protocolului de nivel superior.

Câmpul Lungime/Tip din antetul *Ethernet* este interpretat ca și lungime a cadrului dacă valoarea sa este mai mică de 1536 (0x600 în hexazecimal). Dacă este mai mare de 1536, el reprezintă protocolul de nivel superior folosit.

Câmpul de date trebuie să fie mai mare de 46 de octeți. Dacă din întâmplare datele sunt de lungime mai mică, se adaugă o „umplutură" numită *padding* pentru a ajunge la dimensiunea minimă de 46 de octeți. Acest câmp nu are voie să depășească dimensiunea MTU – *Maximum Transmission Unit* – care pentru *Ethernet* este de 1500 octeți, ceea ce înseamnă că un cadru *Ethernet* nu are voie să fie mai mic de 64 sau mai mare de 1518 octeți.

[1] http://standards.ieee.org/regauth/oui/

Material cu drept de autor

Suma de control este ataşată la sfârşitul cadrului, sub forma unui câmp de 4 octeţi, în scopul detectării erorilor de transmisie. Numărul erorilor CRC este extrem de redus în reţele locale actuale, astfel că relevanţa acestui câmp este scăzută.

Succesul viitor al reţelelor *Ethernet* pare incontestabil. Standardul *Ethernet* a continuat să se dezvolte şi este încă în curs de dezvoltare. În reţelele locale preţurile switchurilor de Gigabit*Ethernet* şi a interfeţelor de reţea se apropie tot mai mult de preţurile de Fast*Ethernet*.

Metro *Ethernet* a devenit o tehnologie răspândită în reţelele de ISP, tehnologie ce îşi propune aducerea standardului *Ethernet* în reţelele metropolitane (MAN), nucleul reţelei fiind bazat pe o structură IP/MPLS deja existentă. Noi standarde la 10 Gbps pe cupru şi fibra optică *multi-mode* au apărut deja pentru a răspunde cerinţelor de bandă din reţelele metropolitane. Prin mărirea vitezelor şi a lungimii maxime a unui segment, *Ethernet* devine o tehnologie viabilă atât pentru MAN cât şi pentru WAN.

În figura de mai jos este prezentată o evoluţie sumară a standardelor *Ethernet*. În iulie 2003 s-a standardizat 802.3ae, ce oferă viteze de până la 10 Gpbs doar pentru reţele de fibră optică. Ultima standardizare a *Ethernet*-ului (IEEE 802.3an) a fost aprobată pe 17 iulie 2006 şi a fost publicată pe 1 septembrie 2006, oferind viteze de până la 10 Gpbs pe cablu torsadat (UTP). Deşi există echipamente de reţea spre vânzare, standardul pentru 40Gbps este aşteptat în 2009.

	Ethernet	Concurenţa
1983	802.3 – 1 Mbps	
		802.5 – 4 Mbps
	802.3 – 10 Mbps	
		802.5 – 16 Mbps
1995		ATM – 155 Mbps
	802.3u – 100 Mbps	
1996		ATM – 622 Mbps
1998	802.3z – 1000 Mbps	
2003	802.3ae – 10 Gbps	

2-2: Evoluţia standardelor *Ethernet*

2.1.2 CSMA/CD

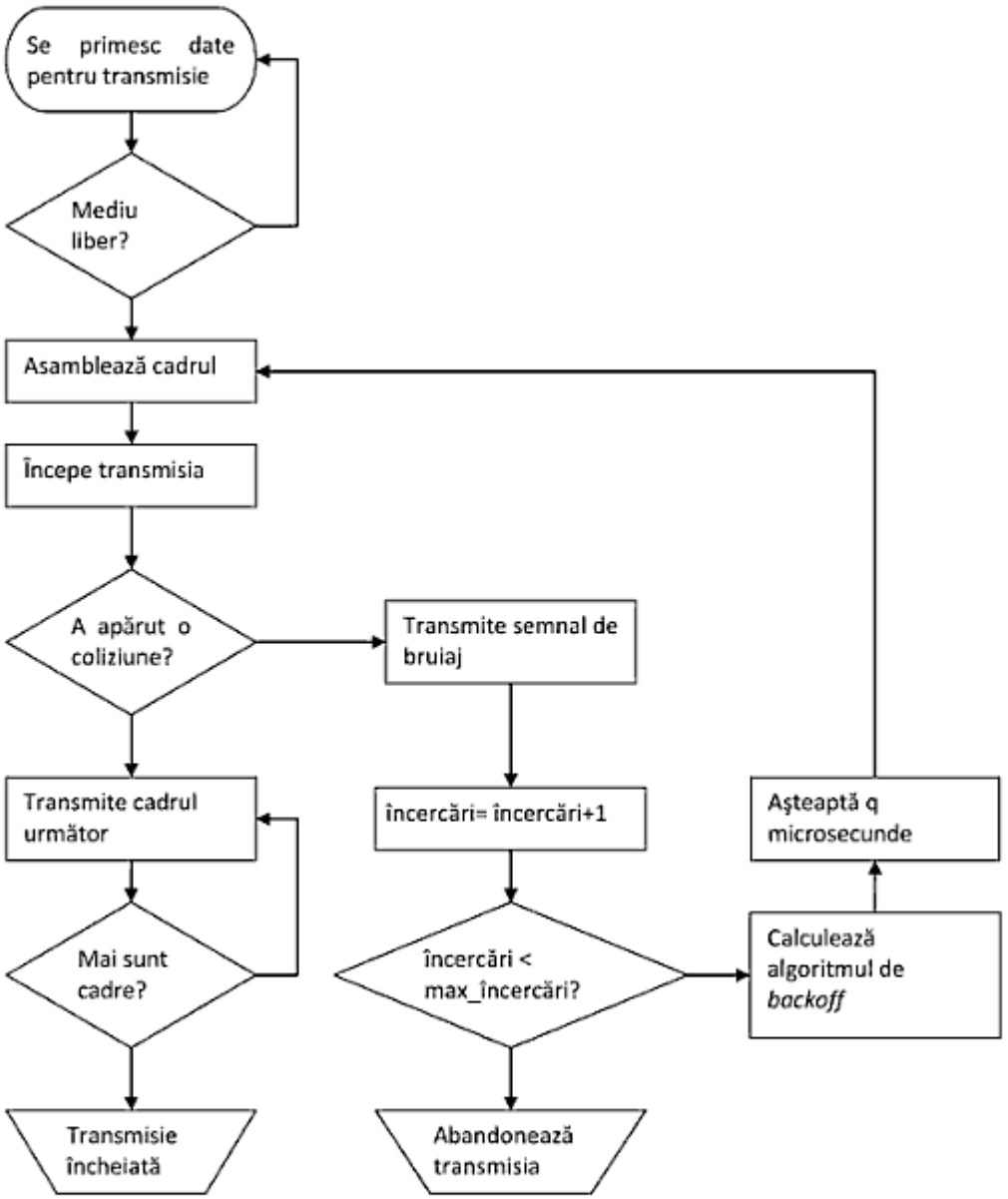
Problema iniţială de la care au pornit reţelele *Ethernet* era găsirea unei metode de arbitrare a accesului la mediul de comunicaţie comun. Această dificultate a fost prima oară explorată în anii `70, la o universitate din Hawaii, într-un proiect ce îşi propunea să ofere acces mai multor utilizatori la o reţea fără ca semnalele lor să se amestece. Rezultatul proiectului a fost reţeaua Alohanet, care a devenit mai târziu baza pentru CSMA/CD (*Carier Sense Multiple Access / Colison Detection*), metoda de acces la mediu folosită în reţelele *Ethernet*.

Reţelele *Ethernet* sunt de tip *shared-media*, deci orice cadru transmis de către o staţie va fi recepţionat de către toate celelalte staţii din reţeaua locală. Toate calculatoarele, la recepţionarea unui cadru valid, vor verifica dacă adresa MAC înscrisă în cadrul câmpului destinaţie din antetul cadrului primit este identică cu adresa MAC proprie. Dacă nu se stabileşte că cele două adrese sunt identice, cadrul este ignorat şi nu va fi transmis către nivelul reţea.

Prezenţa adresei sursă în cadru se explică prin faptul că orice comunicaţie este bidirecţională, în sensul că orice cadru transmis are de obicei ca urmare emiterea unui cadru de răspuns.

Material cu drept de autor

Protocolul CSMA/CD este cel pe baza căruia funcţionează *Ethernet*. Deoarece *Ethernet* se bazează pe un mediu de tip partajat (*shared-media*), numai o singură staţie poate transmite la un moment dat.



2-3: Tratarea coliziunilor

Când o staţie doreşte să transmită, ea urmează următorul procedeu: „ascultă" pe fir să vadă dacă nu cumva mai transmite cineva în acel timp. Dacă „aude" că mai transmite cineva, staţia aşteaptă o perioadă de timp aleatoare după care încearcă din nou (adică ascultă din nou). Dacă nu transmite nimeni, staţia începe să transmită, însă în acelaşi timp continuă şi să

Material cu drept de autor

asculte, pentru a fi sigură că nu a mai început nimeni să transmită. După ce transmisia se termină, stația se întoarce la starea inițială în care ascultă.

Se poate însă ca două stații, urmând acest procedeu, să constate că nu mai transmite nimeni, și considerând mediul liber, să înceapă transmisia simultan, moment în care cadrele de la două stații diferite sunt pe același mediu, ceea ce determină o coliziune.

O coliziune are loc atunci când doi biți de la două stații diferite care transmit se află pe același mediu de transmisie în același timp.

În cazul transmisiunilor analogice pe cupru (cablu coaxial), tensiunile celor două semnale binare se adună, generând astfel un al treilea nivel de tensiune. Această variație a tensiunii este interpretată drept o coliziune.

În cazul transmisiunilor digitale (cablu torsadat sau fibră optică), mediul de transmisie este *full-duplex*, iar coliziunile apar la nivelul emițătorului, a receptorului sau a dispozitivelor de interconectare (spre exemplu un hub va permite apariția de coliziuni). Coliziunile apărute pe o placă de rețea sau pe un switch *half-duplex* sunt coliziuni logice. Astfel dacă o stație începe să trimită un cadru, dar înainte de a finaliza transmisia începe să recepționeze trafic, se va interpreta situația ca și o coliziune, va întrerupe transmisia și va iniția algoritmul de *backoff*.

Când stațiile „aud" coliziunea, continuă să transmită încă o perioada foarte scurtă de timp (semnalul de *jam*) pentru a fi sigure că toate stațiile din rețea au sesizat-o. După ce această coliziune a fost remarcată de toate stațiile din rețea (mai exact, din domeniul de coliziune), este apelat un algoritm de *backoff* și transmisia încetează. Toate stațiile se opresc din transmis pentru o perioadă aleatoare de timp, după care reîncearcă să transmită.

2.1.3 Full-duplex Ethernet

Există două moduri de transmisie (numite și duplex): *half-duplex* și *full-duplex*. În modul de transmisie *half-duplex* o stație nu poate trimite și primi în același timp: ori transmite, ori primește.

De exemplu, cablul coaxial este prin definiție un mediu pentru *half-duplex*, pentru că transmisia și recepția se realizează pe același fir. Pe cablurile torsadate însă, prin folosirea unei perechi separate pentru transmisie (numită Tx) și unei alte perechi pentru recepție (numită Rx) se poate asigura suport de comunicație full-duplex.

Pentru ca o comunicație să fie full-duplex trebuie ca atât mediul de transmisie, cât și participanții la conversație să ofere suport full-duplex.

Cablul torsadat reprezintă un mediu de comuncație *full-duplex*, dar dacă se folosește un hub, sau o placă de rețea configurată *half-duplex*, comunicația se va realiza *half-duplex*.

Switchurile și plăcile de rețea actuale oferă suport atât pentru comunicație full-duplex, cât și pentru comunicația half-duplex. Modul de funcționare poate fi configurat manual (din driverul plăcii de rețea, sau interfața switchului), sau automat, în urma negocierii. Dacă autonegocierea nu reușește și nu sunt făcute setări manuale, transmisia se va face pe half-duplex.

Atunci când folosim *full-duplex* nu se mai folosește modul de acces la mediu CSMA/CD pentru că aceasta nu mai este o rețea de tip mediu partajat. Mai exact, dacă este posibilă transmiterea și primirea de date în același timp, nu mai pot avea loc coliziuni. Astfel, în cadrul unei transmisii *full-duplex*, nu se mai realizează detecția de coliziuni.

Limitările de distanță din standardele *Ethernet* sunt specificate pentru modul de acces la mediu CSMA/CD, și nu mai sunt valabile pentru *full-duplex*. Aceste limitări, cel mai adesea, sunt calculate astfel încât să permită tuturor stațiilor conectate să sesizeze apariția unei

Material cu drept de autor

coliziuni. În condiţiile *full-duplex*, singura limitare a distanţei este cea tehnologică. Un exemplu este *FastEthernet* pe fibra optică *single-mode*, a cărui distanţă este limitată de standardul IEEE 802.3 la 3000m, în condiţii de acces CSMA/CD, însă pentru o legătură punct-la-punct *full-duplex*, se pot folosi *transceiver*-uri puternice obţinând o legătură de până la 120km.

Standardul 10Gigabit nu mai prevede un mod de comunicaţie *half-duplex* ci doar *full-duplex*, motiv pentru care toţi parametrii legaţi de apariţia coliziunilor (număr de încercări, timp de *backoff*, etc) sunt nespecificaţi.

2.2 Ethernet switching

În continuare vor fi analizate reţelele *Ethernet* şi *switching*-ul cadrelor la nivel doi.

Switching-ul de nivel doi presupune folosirea adreselor fizice a staţiilor dintr-un LAN în vederea segmentării reţelei. După cum se ştie, *Ethernet* se bazează pe un mediu de tip partajat (*shared-media*), deci numai o singură staţie poate transmite la un moment dat. Se poate ca două staţii să înceapă transmisia simultan, moment în care cadrele de la cele două staţii diferite se află pe acelaşi mediu, ceea ce determină o coliziune. Reţelele *Ethernet* ar funcţiona foarte bine în condiţii ideale. Dacă numărul utilizatorilor în reţea devine însă foarte mare, atunci numărul coliziunilor ar creşte semnificativ, ducând la o scădere a performanţelor reţelei. Astfel, nevoia de a sparge domeniile mari de coliziune în domenii mai mici a fost vitală. Domeniile de coliziune şi cele de difuzare au început să fie treptat proiectate cu scopul de a limita efectul negativ al coliziunilor şi al difuzărilor asupra performanţelor reţelei.

Un domeniu de coliziune *reprezintă acea secţiune dintr-o reţea în care se va propaga o coliziune. Switchurile şi routerele limitează domeniile de coliziune, dar repetoarele le extind.*

Un domeniu de difuzare *(domeniu de broadcast) reprezintă acea secţiune dintr-o reţea în care se va propaga un pachet de difuzare (broadcast). Routerele limitează domeniile de difuzare, dar repetoarele şi swichurile le extind.*

Un segment de reţea *este echivalent cu un domeniu de coliziune, după cum unei reţele îi corespunde un domeniu de difuzare.*

Care sunt mecanismele de decizie ale unei switch? Cele două mecanisme ce fac din switch un dispozitiv de interconectare „inteligent" sunt: încapsularea datelor la nivel legătură de date şi folosirea unei scheme de adresare pentru livrarea acestora. Toate deciziile luate de un switch sunt bazate pe adresa fizica (MAC), neafectând adresa logică (adresa de nivel trei) a pachetului.

Gruparea datelor nu se face la nivel de bit ci la nivel de cadru, un cadru putând conţine până la 1500 de octeţi în cazul cadrului *Ethernet*, 2300 pentru WLAN, sau chiar 8000 de octeţi în cazul *Token Ring*.

În prezent, o reţea locală este formată din switchuri, AP-uri şi routere.

Definiţia cea mai răspândită a switchurilor identifică orice *bridge* multiport cu un **switch**. În realitate, deşi această definiţie acoperă vasta majoritate a cazurilor, există *bridge*-uri multiport ce nu sunt switchuri.

În timp ce switchul este definită ca un dispozitiv de interconectare de nivel legătură de date, definiţia switchului include atât funcţii de nivel fizic, cât şi de nivel legătură de date. Aceasta diferenţă nu se datorează unei latenţe mai mici sau unui cost mai scăzut comparativ cu o punte, ci faptului că în reţelele *Ethernet* ce folosesc mediul torsadat switchul preia funcţia principală a hubului, şi anume aceea de a asigura conectarea tuturor nodurilor la un mediu de transmisie.

Material cu drept de autor

Pe interfețele unui switch se pot conecta stații sau segmente întregi. Cu toate acestea, rețelele *switched* sunt răspunsul la cerințele crescânde de securitate și de lățime de bandă pentru fiecare nod. Rețelele comutate vor folosi câte un port pentru fiecare stație, reducând dimensiunea domeniilor de coliziune la doar două noduri (unul fiind placa de rețea din respectiva stație, iar cel de-al doilea portul din switch care o conectează pe aceasta).

Principalele funcții ale unui switch sunt: **învățarea adreselor** (popularea tabelei MAC) și **modul de comutare a cadrelor**. Pe lângă acestea, există o serie de funcții pe care unele switchuri le pot oferi: eliminarea buclelor de nivel doi, separarea rețelelor locale virtuale, etc.

2.2.1 Învățarea adreselor

Switchurile de nivel doi rețin adresa fizică sursă a cadrului primit pe o interfață și introduc această informație într-o tabelă - **tabelă de comutare** - numită și tabelă MAC (*bridging/switching table*). În această tabelă fiecarei adrese fizice îi este asociată una dintre interfețele sale. În figura de mai jos este exemplificată o astfel de tabelă.

Interfață	Adresă MAC
E0	00.48.C2.01.78.12
E0	00.00.2E.00.59.91
E1	00.00.54.91.01.4A

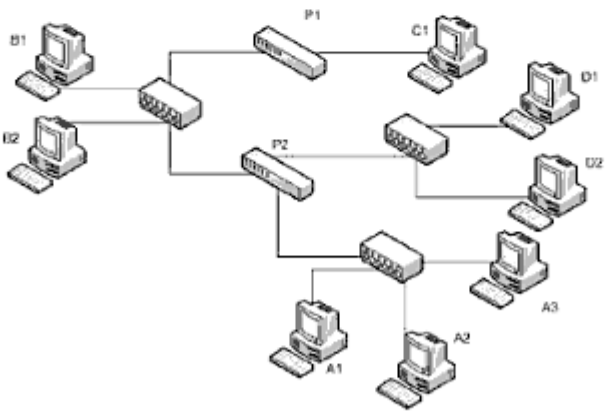
2-4: Tabelă de comutare cu 3 intrări

De exemplu, prima intrare are următoarea semnificație: destinația 00.48.C2.01.78.12 se află pe segmentul conectat pe interfața E0 a switchului (E0 este prescurtarea de la *Ethernet* 0, prima interfață *Ethernet*).

Cum își construiește switchul tabela de comutare?

Tabela de comutare este păstrată în memoria RAM a switchului și prin urmare se pierde dacă switchul este reinițializat. În plus, un switch trebuie să includă dinamic în tabela de comutare informații despre o nouă stație conectată în rețea.

Se consideră rețeaua din figura de mai jos, unde switchul 1 a fost reinițializat, și stația A1 vrea să comunice cu stația B1.



2-5: Construirea tabelei de comutare pentru o rețea cu switchuri

Stația A1 ascultă mediul, iar când acesta este liber trimite un cadru ce are ca destinație stația B1. Stațiile A2 și A3 ignoră cadrul. Switchul 1 primește cadrul și încearcă să găsească adresa destinație în tabela sa de comutare. Switchul nu reușește să găsească destinația,

Material cu drept de autor

deoarece tabela sa de comutare este goală, astfel încât el transmite cadrul pe toate segmentele la care este conectat, în afară de segmentul de pe care a fost primit cadrul (*flooding*). În această etapă, funcţionarea switchului este similară cu cea a unui hub. Înainte de a retransmite cadrul switchul verifică dacă adresa sursă este prezentă în tabela sa de comutare. În acest caz nu este, astfel încât switchul creează prima intrare în tabela de comutare, intrare ce conţine adresa fizică a staţiei A1 şi interfaţa switchului ce conectează segmentul A. Cadrul ajunge atât pe segmentul D (unde staţiile D1 şi D2 decid că acesta nu le este adresat şi îl ignoră) cât şi pe segmentul B, la staţiile B1 şi B2 şi la switchul 2.

Switchul 2 decide că destinaţia se află în segmentul din care a primit cadrul şi deci nu îl mai retransmite, iar staţia B1 decide că ea este destinatarul cadrului.

Următoarele intrări în tabela de comutare sunt adăugate în mod similar, pe baza informaţiilor conţinute în câmpul sursă a cadrelor ce ajung la switch.

Chiar şi comunicaţia între două staţii aflate în acelaşi segment poate afecta lăţimea de bandă din întreaga reţea, dacă switchul nu a apucat să-şi construiască tabela de comutare.

Se consideră că staţia A1 trimite un cadru către A2, după cel trimis către B1. Cadrul ajunge la destinaţie fără ajutorul switchului, dar switchul, neidentificând destinaţia în tabela sa de comutare, va retransmite cadrul atât pe segmentul B, cât şi pe segmentul D.

Datorită dificultăţii căutării într-o mulţime neordonată, în tabela de comutare nu sunt păstrate toate adresele staţiilor din reţeaua locală, ci doar ale celor cu o probabilitate mare să transmită în viitorul apropiat - mai exact a ultimelor staţii care au transmis. Pentru implementarea acestui concept, o intrare într-o tabelă de comutare va include şi o etichetă de timp, pe lângă adresa MAC şi interfaţă. Eticheta de timp este actualizată când se primeşte un nou cadru cu aceeaşi adresă sursă. Acest mecanism permite înlăturarea intrărilor învechite şi duce deci la restrângerea dimensiunii tabelei de comutare. Preţul plătit, în cazul în care o staţie nu transmite niciun cadru un interval de timp, este consumul din lăţimea de bandă a tuturor segmentelor din reţea.

Există o singură excepţie notabilă la procesul descris mai sus: în cazul în care adresa destinaţie a cadrului este aceeaşi cu adresa sursă, cadrul este considerat invalid, va fi aruncat, în plus adresa sursă nu va fi folosită pentru popularea tabelei de comutare.

2.2.2 Deciziile de comutare

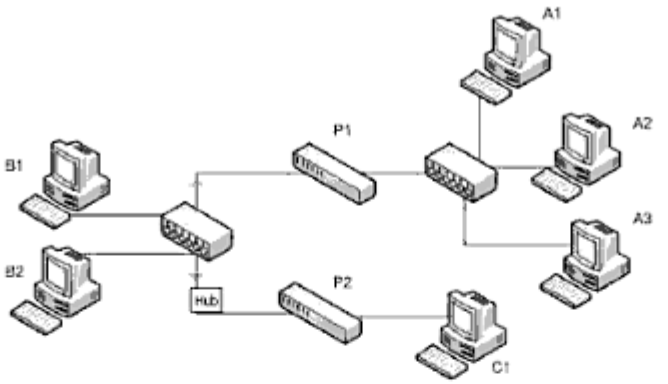
Care este rolul switchului în comunicaţia din interiorul aceluiaşi segment?

Protocolul *Ethernet* oferă un mediu de comunicaţie partajat, mai exact comunicaţia dintre două staţii este accesibilă nivelului legătură de date a oricărei alte staţii conectate pe acelaşi segment. Pentru fiecare cadru primit de o staţie, nivelul legătură de date verifică dacă această staţie este sau nu destinaţia. În cazul afirmativ cadrul este pasat nivelului reţea, altminteri este ignorat.

Reţeaua din figura de mai jos ilustrează cazul comunicaţiei în interiorul aceluiaşi segment. De exemplu, staţia A1 vrea să transmită date staţiei A2.

Deoarece este vorba despre o reţea *Ethernet*, primul lucru pe care-l face staţia A1 este ascultarea mediului. Dacă mediul este liber începe transmisia datelor. Cadrul emis de A1 se propagă către toate staţiile conectate pe acest segment, inclusiv către switch. Staţia A2 trece cadrul către nivelul reţea, staţia A3 îl ignoră. Odată ajuns la switch, cadrul este despachetat şi adresa destinaţie este căutată în tabela de comutare a switchului. Switchul decide că destinaţia se află chiar pe interfaţa pe care a primit cadrul. În acest caz switchul ia decizia că acest cadru nu mai trebuie transmis (*filtering*), deoarece retransmiterea cadrului ar duce la o duplicare a acestuia la destinaţie.

Material cu drept de autor



2-6: Reţea segmentată cu switchuri

Cum acţionează switchul 1 în cazul comunicaţiei între B1 şi B2? Ambele switchuri (deşi recepţionează cadrele) iau decizia de a nu le mai retransmite.

Dacă ambele comunicaţii apar simultan (atât A1 transmite către A2, cât şi B1 către B2), va apărea oare o coliziune? Cu siguranţă că da, dacă în loc de switchul 1 ar fi fost folosit un repetor. În cazul utilizării unui switch, de vreme ce niciun cadru din comunicaţia dintre A1 şi A2 nu ajunge pe segmentul B, şi niciun cadru din comunicaţia dintre B1 şi B2 nu ajunge pe segmentul A, este imposibil să apară o coliziune.

Switchul izolează comunicaţia unicast între staţii aflate în acelaşi segment la nivelul segmentului.

Consecinţele acestui fapt sunt extrem de importante. În primul rând, switchul mărgineşte domeniile de coliziune. Totodată, el oferă mai multă bandă disponibilă, deoarece comunicaţia în interiorul aceluiaşi segment nu consumă din banda disponibilă a întregii reţele.

O altă consecinţă o reprezintă minimizarea riscurilor de securitate legate de atacurile din interiorul reţelei locale. De exemplu, unul dintre cele mai populare atacuri este ascultarea liniei (*sniffing attack*), prin care se forţează nivelul legătură de date de pe una dintre staţiile conectate la mediul distribuit să trimită spre nivelurile superioare toate cadrele - inclusiv cele ce nu sunt destinate acestei staţii. Cu ajutorul unor aplicaţii dedicate datele sunt reasamblate şi astfel poate fi monitorizat tot traficul ce traversează segmentul de reţea. Dar, prin folosirea switchurilor, este posibil ca staţiile ce prezintă un risc de securitate să fie izolate de restul reţelei.

Care este rolul switchului în comunicaţia dintre segmente?

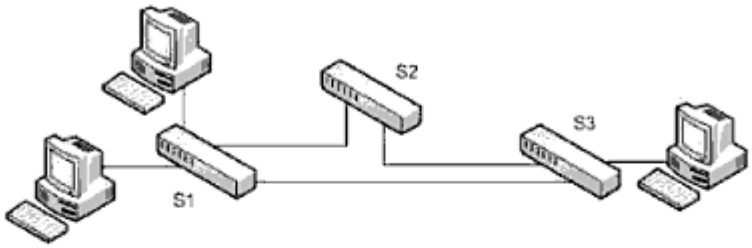
Dacă în reţeaua din figura precedentă este iniţiat un trafic între staţia A1 şi B1, staţia A1 începe prin a asculta mediul şi transmite un cadru atunci când acesta este liber. Cadrul se propagă spre staţiile A2, A3 şi spre switchul 1. Staţiile ignoră cadrul, acesta nefiind adresat lor. Switchul căută însă adresa destinaţie în tabela sa de comutare. Switchul determină interfaţa pe care trebuie trimis cadrul şi apoi decide că această interfaţă este diferită de cea pe care cadrul a fost primit. Astfel, switchul transmite cadrul primit din segmentul A doar pe segmentul B (*forwarding*). Switchul este recepţionat atât de B1, cât şi de B2, dar doar B1 îl va prelucra.

2.2.3 Evitarea buclelor de nivel doi – STP

Una dintre principalele limitări ale nivelului legătură de date se referă la posibilitatea asigurării redundanţei. Astfel, pentru început va fi analizat impactul redundanţei asupra reţelelor formate numai din switchuri.

Material cu drept de autor

O **buclă de nivel legătură de date** apare într-o reţea atunci când între două dispozitive ale acesteia există două sau mai multe legături active, fiecare conexiune folosind doar dispozitive de interconectare ce pot analiza cel mult informaţii de nivel legătură de date.



2-7: Reţea în care s-a creat o buclă

Care este efectul apariţiei buclelor de nivel legătură de date?

Apariţia buclelor de nivel *legătură de date* este corelată cu faptul că switchurile nu filtrează pachetele de difuzare şi duc la o depreciere semnificativă a performanţelor reţelei prin determinarea unei **avalanşe de difuzări** (*broadcast storm*).

Se consideră reţeaua din figura de mai sus. Se presupune că staţia A trimite un cadru de difuzare. Switchul 1 nu va găsi adresa destinaţie în tabela sa de comutare, astfel încât va transmite cadrul pe celelalte segmente: segmentul ce conţine staţia B, segmentul dintre switchurile 1 şi 2 precum şi segmentul dintre switchurile 1 şi 3. Staţia B examinează cadrul, decide că îi este adresat şi îl trece spre nivelul legătură de date. Switchul 2 ia decizia de a transmite cadrul pe toate interfeţele sale, cu excepţia celei de pe care a primit cadrul. Astfel apar în reţea două cadre destinate staţiei FF.FF.FF.FF.FF.FF, adică două cadre de difuzare. Indiferent de ordinea în care acestea ajung la switchul 3, acesta va decide că nu cunoaşte adresa destinaţie şi le va retransmite către staţia C, dar şi către celelalte switchuri.

Avalanşa de difuzări consumă din banda utilă a reţelei, ducând la o micşorare a bandei efective disponibile. O avalanşă de difuzări se opreşte doar în cazul întreruperii buclei.

Cum poate fi prevenită apariţia avalanşelor de difuzări?

Soluţia trivială este ca switchurile să fie instruite să nu retransmită cadrele de difuzare. Din păcate acest lucru este imposibil, deoarece o serie de protocoale folosesc cadre de difuzare pentru a funcţiona corect, unul dintre acestea fiind chiar ARP - *Address Resolution Protocol*. Altfel spus, filtrarea cadrelor de difuzare de către switchuri ar presupune rescrierea protocoalelor fundamentale ce asigură suportul de comunicaţie.

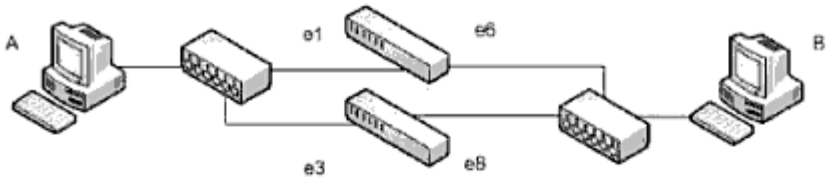
Soluţia validă se bazează pe identificarea buclelor şi întreruperea lor. Protocolul ce realizează aceasta se numeşte STP - *Spanning Tree Protocol*, şi porneşte de la construirea unui **arbore de acoperire** pe graful determinat de dispozitivele de interconectare şi de conexiunile dintre acestea.

Sunt problemele de redundanţă legate exclusiv de cadrele de difuzare?

Cadrele de difuzare sunt principala cauză a deformării performanţelor unei reţele cu bucle de nivel legătură de date. Cu toate acestea, redundanţa poate duce la învăţarea de informaţii false chiar în absenţa cadrelor de difuzare.

Pentru a vedea comportamentul unei reţele redundante în cazul reiniţializării unuia dintre switchuri, se va considera reţeaua din Figura 4-8. Astfel staţia A, interfaţa e1 din switchul 1 (care va fi notat în continuare S1) şi interfaţa e3 din switchul 2 (S2) fac parte din acelaşi domeniu de coliziune. În mod asemănător staţia B, interfaţa e6 din S1 şi interfaţa e8 din S2 partajează acelaşi mediu de comunicaţie.

Material cu drept de autor



2-8: Rețea de switchuri cu o buclă

S1 a fost reinițializat, astfel încât când stația A va trimite un cadru destinat stației B, S1 nu va găsi nicio informație în tabela sa de comutare despre B. În acest caz va introduce în tabela sa de comutare corespondența între interfața sa e1 și adresa MAC a stației A. În paralel cu actualizarea tabelei de comutare S1 va trimite cadrul pe toate celelalte interfețe în afară de e1, inclusiv pe interfața e6. Astfel, cadrul va ajunge atât la destinație, cât și pe interfața e8 din S2. S2, știind că destinația (stația B) se află pe același segment cu interfața e8 va ignora cadrul.

Mediu de comunicație fiind partajat cadrul transmis de A va ajunge atât la S1, cât și la S2. Cadrul primit pe interfața e3 va fi procesat de S2 și trimis pe interfața e8. Astfel cadrul va ajunge încă odată la destinație, stația B primind două copii ale aceluiași cadru. În plus cadrul va ajunge și la switchul 1, care după ce își va inspecta tabela de comutare, va găsi corespondența între adresa MAC a stației A și e1. Cadrul semnat cu adresa MAC a stației A fiind primit pe interfață e6, switchul consideră că informațiile din tabela de comutare sunt alterate și va invalida această intrare, creând apoi o nouă intrare ce va conține corespondența între adresa MAC a stației A și interfața e6. În plus, cadrul va fi trimis pe toate interfețele în afară de e6, inclusiv pe e1.

S-a ajuns astfel în situația în care cadrul de *unicast* trimis de stația A va fi livrat la destinație de un număr nelimitat de ori, în plus tabela switchului 1 ștergând și adăugând alternativ corespondențele între adresa MAC a stației A și interfețele e1 pe de o parte și corespondența cu e6 pe de altă parte.

Până în acest punct s-a argumentat faptul că redundanța la nivelul legătură de date implică asumarea unor riscuri importante. Soluția pentru asigurarea redundanței rețelei locale constă în păstrarea redundanței de nivel fizic, dar întreruperea buclelor de nivel legătură de date, această soluție fiind implementată de STP (*Spanning Tree Protocol*).

Cum funcționează STP?

Funcționarea acestui protocol se bazează pe crearea topologiei rețelei folosind niște cadre speciale, numite cadre BPDU (*Bridge Protocol Data Unit*). Aceste cadre speciale sunt folosite intens la inițializarea switchurilor; ulterior, la fiecare două secunde sunt schimbate cadre BDPU, pentru a verifica dacă nu au apărut modificări. Totodată sunt definite cinci stări în care se poate afla o interfață a switchului: starea **blocat, de ascultare, de învățare, de comutare de cadre și nefuncțional** (*blocking, listening, learning, forwarding, disabled*). În starea blocat nu se acceptă decât cadre BPDU, în cea de ascultare se primesc și cadre, dar acestea nu sunt retransmise. În starea de învățare, în plus față de starea de ascultare, este inspectată adresa sursă a cadrelor primite, permițând astfel construirea tabelei de comutare. În starea de comutare, cadrele primite sunt retransmise, iar tabela de comutare este actualizată. În starea nefuncțional nu sunt acceptate nici cadre BPDU.

Pentru construirea arborelui de acoperire sunt necesare aproximativ 50 de secunde, timp în care toate porturile switchurilor sunt în starea blocat. Există trei pași ce trebuie urmați pentru construirea arborelui de acoperire: mai întâi trebuie aleasă **rădăcina arborelui** (*root bridge*), apoi trebuie alese **porturile rădăcină**, pentru ca în final să fie stabilite **porturile active**.

Material cu drept de autor

Prioritatea switchului este o valoare numerică păstrată în memoria nevolatilă a fiecărui switch. Pe baza comparării priorităţilor tuturor switchurilor din reţea, este identificat switchul cu prioritatea cea mai scăzută, aceasta devenind rădăcina arborelui de acoperire.

Prioritatea switchului are o valoare implicită atribuită de producător, valoare ce poate fi modificată ulterior. În cazul folosirii mai multor echipamente produse de aceeaşi firmă, se întâmplă adesea să existe mai multe switchuri cu aceeaşi prioritate. Cum poate fi stabilit care dintre două sau mai multe switchuri cu aceeaşi prioritate să devină rădăcina arborelui? Pe baza adresei fizice: switchul cu cea mai mică adresă fizică devine rădăcina arborelui de acoperire.

Pasul al doilea presupune identificarea căilor redundante dintre fiecare switch şi switchul rădăcină, apoi selectarea unei sigure căi între respectivul switch şi rădăcină şi, în final, dezactivarea celorlalte. Pentru aceasta trebuie decis căreia dintre cele trei categorii de porturi: port rădăcină (RP), port activ (DP – *designated ports*) sau port inactiv (*nondesignated ports* – NP) îi va aparţine fiecare dintre porturile ce participă la STP.

Pentru evaluarea unei căi este calculat *costul căii*, cost ce este definit ca sumă a costurilor porturilor prin care trece calea. Costul unui port este definit la rândul său în funcţie de lăţimea de bandă pe care o oferă portul.

Lăţime de bandă	Cost
4 Mbps	250
10 Mbps	100
16 Mbps	62
45 Mbps	39
100 Mbps	19
155 Mbps	14
622 Mbps	6
1 Gbps	4
10 Gbps	2

2-9: Costul STP în funcţie de lăţimea de bandă

De-a lungul timpului au existat mai multe metode de calcul a costului unui port. Cu câţiva ani în urmă costul portului pentru switchurile CISCO era determinat împărţind 1000 la lăţimea de bandă pe care o oferea portul, astfel încât un port *Ethernet* avea costul 100.

Pentru alegerea porturilor rădăcină au prioritate porturile conectate direct la rădăcina arborelui de acoperire. În cazul în care nu există niciun port cu o conexiune directă spre switchul rădăcină, sau când există mai mult de un singur port cu conexiune directă spre rădăcină, este ales portul care are cel mai scăzut cost al căii spre rădăcină, urmând ca acest port să fie marcat ca port rădăcină.

Ultimul pas din construcţia arborelui de acoperire presupune determinarea porturilor active. Un port activ este un port ce trimite şi recepţionează trafic, în vreme ce un port inactiv este trecut în starea blocat.

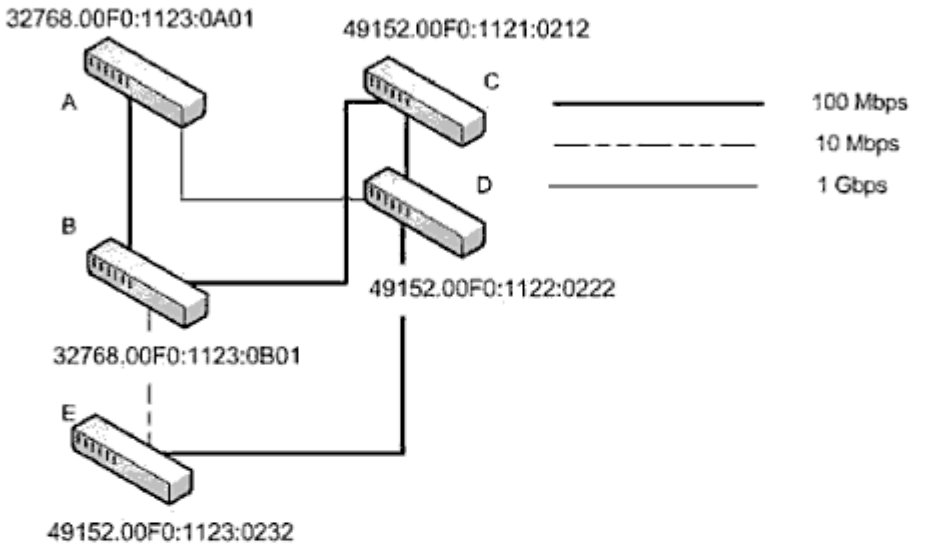
Odată stabilite porturile rădăcină, trebuie identificate, pe de o parte, conexiunile ce fac parte dintr-o cale optimă către switchul rădăcină (deci care au la unul dintre capete un port marcat ca port rădăcină), şi, pe de altă parte, cele ce nu aparţin din arborele de acoperire (deci conexiunile care trebuie întrerupte).

În primul caz metoda este evidentă: toate porturile ce fac parte dintr-o cale optimă şi nu sunt deja marcate ca porturi rădăcină sunt marcate drept porturi active.

Material cu drept de autor

În cel de al doilea caz trebuie făcută o comparaţie între cele două porturi ce definesc conexiunea redundantă. Mai întâi sunt comparate costurile celor două porturi; dacă acestea sunt egale, factorul decisiv este identificatorul switchului: portul de pe switchul cu identificatorul cel mai mic (cu prioritatea cea mai mică sau în cazul în care priorităţile sunt egale, cu adresa MAC cea mai mică) devine port activ, în vreme ce portul cu identificatorul mai mare este marcat ca inactiv şi este trecut în starea blocat.

Fie reţeaua din figură. Se vor urmări pentru această reţea cu două bucle etapele construirii arborelui de acoperire.



2-10: Construirea arborelui de acoperire

Prima întrebare este: care este rădăcina arborelui de acoperire? Pentru aceasta sunt comparate priorităţile celor 5 switchuri. În cazul reţelei de mai sus pornim de la premisa că toate switchurile sunt produse de acelaşi fabricant şi în plus sunt abia scoase din cutie. Altfel spus, toate switchurile au aceeaşi prioritate. În acest caz trebuie comparate adresele fizice.

Switchul cu adresa MAC cea mai mică este A. Astfel A devine rădăcina arborelui de acoperire.

La pasul al doilea trebuie stabilite pentru restul switchurilor costurile porturilor ce oferă căi spre switchul rădăcină. Pentru o legătură *Ethernet* costul e 100, pentru una Fast*Ethernet* e 19, iar pentru una Giga*Ethernet* e 4; prin urmare:

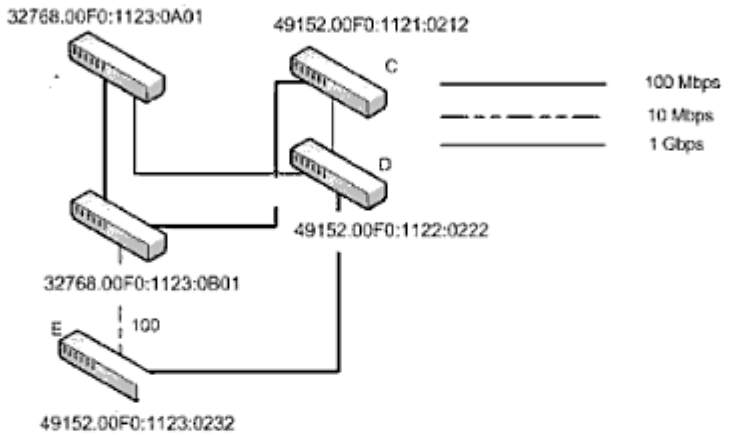
B-Root: 19
B-C-D-Root: 42
B-C-D-E-B-Root: 176
C-B-Root: 38
C-D-Root: 23
C-D-E-B-Root: 157
D-Root: 4
D-C-B-Root: 57
D-E-B-Root: 138
E-B-Root: 119
E-D-Root: 23
E-D-C-B-Root: 76

Material cu drept de autor

Conexiunile ce fac parte din arborele de acoperire sunt: B-A, C-D-A, D-A şi E-D-A, astfel sunt marcate drept porturi rădăcină cele patru porturi corespunzătoare acestor conexiuni (de pe switchul B portul ce face parte din legătura B-A, etc.)

În pasul al treilea trebuie stabilite porturile active pentru legăturile B-E şi B-C.

Pentru conexiunea B-E, costul portului de pe switchul B este 19, în vreme ce costul portului de pe E este 23. Prin urmare portul de pe B va fi marcat ca port activ, iar portul de pe E va fi marcat ca *non-designated* sau inactiv. În mod similar, pentru conexiunea B-C, portul de pe switchul B va fi activ, având un cost de 19, iar portul de pe C va fi trecut în starea blocat, fiind un port *non-designated*.



2-11: Configuraţia finală

2.2.4 Metode de comutare

Există două metode de comutare a pachetelor: **comutare directă** (*cut through*) şi **comutare după stocare** (*store and forward*).

Metoda de comutare după stocare se bazează pe recepţionarea întregului cadru înainte de a începe retransmisia acestuia. Latenţa acestei metode creşte odată cu dimensiunea câmpului de date. Cu toate acestea, performanţele metodei de comutare după stocare pot fi superioare celor oferite de comutarea directă, mai ales în cazul liniilor expuse unor interferenţe puternice. Mecanismele de detecţie a erorilor pe care le oferă această metodă permit asigurarea unei conexiuni sigure la nivelul legătură de date.

Metoda de comutare după stocare pune şi problema asigurării memoriei pentru stocarea cadrelor. Fie exemplul unui switch cu 24 de porturi. Acesta trebuie să poată gestiona 12 comunicaţii simultane, care în cel mai defavorabil caz posibil vor transfera cadre de lungime maximă. Se ajunge astfel la o dimensionare a memoriei RAM necesară pentru stocarea cadrelor de aproape 18 kB. Deşi dimensionarea memoriei RAM folosite pentru stocarea cadrelor nu este principalul factor de stabilire a preţului unui switch, nu trebuie omis faptul că preţurile pentru memoriile dispozitivelor dedicate sunt de câteva ori mai ridicate decât cele pentru memoriile folosite în calculatoarele personale.

Comutarea directă presupune ca dispozitivul de interconectare să înceapă transmiterea cadrului pe portul destinaţie imediat ce adresa destinaţie a fost trecută prin tabela de comutare şi interfaţa de plecare a fost determinată. Cel mai adesea se întâmplă ca transmisia cadrului să înceapă înainte de recepţionarea integrală a cadrului. Astfel switchul primeşte pe una dintre interfeţe octeţi ce compun cadrul, transmiţând în acelaşi timp pe portul destinaţie octeţii din acelaşi cadru primiţi mai devreme.

Material cu drept de autor

Pentru comutarea directă nu este necesară nici măcar recepționarea integrală a antetului cadrului, adresa destinație fiind suficientă. Această metodă se numește **comutare directă rapidă** (*fast forward*) și oferă o latență de aproximativ 21 de microsecunde. Datorită faptului că retransmisia cadrului începe imediat după citirea adresei destinație, cadrele eronate vor fi transmise cu erori. Deși aceste cadre sunt respinse la nivelul legătură de date al destinației (de către placa de rețea), traficul generat de retransmisia lor poate, în cazul unui mediu de transmisie cu multe erori, să ducă la o depreciere severă a performanțelor rețelei.

Al doilea tip de comutare directă este **comutarea fără fragmente** (*fragment free*). În această metodă de comutare fragmentele de cadre rezultate în urma unei coliziuni sunt filtrate. Într-o rețea ce respectă specificațiile standardului *Ethernet* dimensiunea fragmentelor de coliziuni nu poate depăși 64 de octeți. În cazul comutării fără fragmente, switchul decide că șirul de octeți recepționați nu face parte dintr-un fragment rezultat în urma unei coliziuni și abia apoi începe retransmisia pe portul destinație. Latența în acest caz este de minim 51,2 microsecunde, timpul necesar recepționării a 64 de octeți.

2.2.5 Switch vs. Bridge

Care sunt diferențele dintre un switch și un bridge?

Cele mai importante două diferențe dintre un switch și un bridge se referă la metodele de comutare oferite și la proiectarea *backplain*-ului.

Față de bridge-uri, switchurile implementează în general metode de comutare mai rapide. În general bridge-urile, deși nu sunt interesate de detecția unui număr cât mai mare de erori, implementează doar comutarea după stocare. Această practică este explicabilă mai degrabă prin rațiuni istorice, nefiind rezultatul unei decizii de optimizare a traficului în rețea.

Cea de a doua diferență se referă la capacitatea switchurilor de a permite mai multe comunicații simultane, fără a scădea lățimea de bandă alocată fiecăreia dintre conexiuni. Spre deosebire de un switch, o punte va avea o capacitate de comutare internă (*backplain*) aproximativ egală cu viteza porturilor. Altfel spus, în cazul unui bridge cu 4 porturi de 10 Mbps, pentru o singură conexiune lățimea de bandă disponibilă va fi de 10 Mbps, dar în cazul inițierii a două conexiuni simultane (stația de pe portul 1 comunică cu cea de pe portul 3 și în același timp stația de pe portul 2 comunică cu cea de pe portul 4), fiecare dintre cele două conexiuni va avea o bandă disponibilă de 5 Mbps.

Cele două diferențe dintre switchuri și bridge-uri sunt în fapt avantaje importante ale switchurilor, iar prețul unui switch este foarte apropiat de cel al unui bridge. Cu toate acestea se mai produc bridge-uri și în ziua de azi.

Există un caz în care cele două avantaje ale switchurilor își pierd relevanța. Este vorba despre interconectarea a două rețele ce folosesc protocoale de nivel 2 diferite. În acest caz singura metodă de comutare posibilă este *store-and-forward*, deoarece cel mai adesea cadrele trebuie reîmpachetate, datorită diferențelor de dimensiune maximă a cadrului, dar și datorită diferențelor de format a antetelor. În plus, pentru interconectarea a două rețele cu protocol de nivel legătură de date diferit se folosește în general un dispozitiv dedicat, astfel încât *backplain*-ul dispozitivului să nu trebuiască să facă față la mai mult de o conexiune la un moment dat.

Cărui fapt i se datorează variația de preț între switchuri?

Numărul de porturi este unul din factorii ce determină prețul unui switch. Există un cost mediu pe port, acest cost variind în funcție de viteză și de producător. De exemplu, costul pentru un port *Ethernet* se situează între 8 și 10 $.

Pe de altă parte, datorită numărului mare de servicii pe care le poate oferi un switch, de la simpla comutare de pachete până la rularea STP sau chiar a SNMP, prețurile switchurilor

Material cu drept de autor

variază extrem de mult. Astfel, dacă un switch cu 8 porturi fără nicio facilitate suplimentară poate să coste sub 100$, un switch aparținând aceluiași producător, dar oferind un management avansat și posibilitatea inserării de noi module poate să ajungă la 2000$.

2.3 Rețele locale virtuale

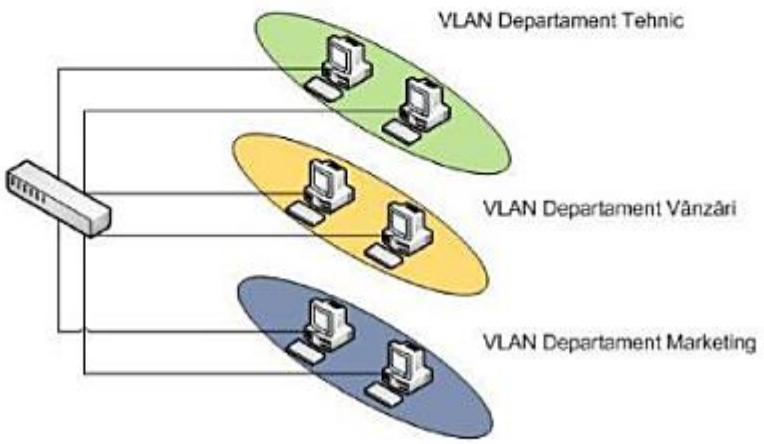
După cum s-a văzut în secțiunea anterioară, folosirea switchurilor într-o rețea *Ethernet* are ca efect segmentarea acesteia în domenii de coliziune individuale, neavând niciun efect asupra domeniului de difuzare. Acest lucru înseamnă că toate nodurile din rețea pot să vadă broadcast-ul trimis de un nod al unui segment.

O caracteristică importantă a comutării în rețelele *Ethernet* o reprezintă abilitatea de a crea LAN-uri virtuale (**VLAN – Virtual LAN**). Conceptul de rețea virtuală a fost standardizat de către comitetul 802 (IEEE), fiind utilizat în ziua de azi de multe organizații. Un VLAN reprezintă o grupare logică a stațiilor/utilizatorilor și echipamentelor de rețea, fără nicio restricție asupra segmentului fizic din care fac parte. VLAN-urile segmentează o rețea comutată, ținând seama de organizarea în echipe de lucru sau de aplicații și nu de criterii geografice. Altfel spus, o rețea locală virtuală poate fi privită ca un domeniu logic de broadcast definit de o mulțime de switchuri.

Traficul între VLAN-uri este restricționat. Switchurile transmit trafic de tip *unicast*, *multicast* și *broadcast* numai pe segmentele de rețea ce fac parte din VLAN-ul de care aparțin. Cu alte cuvinte nodurile dintr-un VLAN comunică numai cu nodurile din același VLAN. Pentru comunicare între VLAN-uri diferite este nevoie de un dispozitiv de nivel trei, și anume un router.

De ce este nevoie de VLAN-uri?

Marele avantaj obținut prin introducerea switchurilor într-o rețea *Ethernet* constă în crearea domeniilor de coliziune independente pentru fiecare port al switchului. Dar, cu fiecare pas, apar noi probleme – cu cât sunt mai mulți utilizatori și/sau mai multe switchuri, cu atât numărul *broadcast*-urilor va fi mai mare. Altfel spus, pe măsură ce din ce în ce mai multe rețele locale sunt interconectate, numărul cadrelor de *broadcast* recepționate de fiecare stație tinde să crească liniar cu numărul de stații.



2-12: Topologie cu trei domenii de broadcast

Toate echipamentele dintr-un VLAN sunt membre ale aceluiași domeniu de *broadcast*, recepționând astfel toate *broadcast*-urile. Toate cadrele de *broadcast* sunt filtrate de toate

Material cu drept de autor

porturile unui switch ce nu sunt în VLAN-ul respectiv. Acesta este un avantaj deoarece oferă toate beneficiile pe care le oferă o reţea comutată, dar fără dezavantajele de a avea toţi utilizatorii într-un singur domeniu de *broadcast*. Cu ajutorul VLAN-urilor putem controla uşor *mărimea domeniilor de broadcast* prin: controlul dimensiunii totale a VLAN-urilor sale; restricţionarea numărului de porturi pe switch în cadrul unui VLAN; restricţionarea numărului de utilizatori ce folosesc aceste porturi.

O altă problemă o reprezintă securitatea. În cadrul unei reţele *Ethernet* comutate toate staţiile pot vedea toate celelalte staţii şi nu există o politică de oprire a *broadcast*-urilor trimise de staţii sau de oprire a răspunsurilor utilizatorilor la ele. Orice placă de reţea poate fi setată într-un mod transparent (*promiscuous*), copiind tot traficul ce soseşte pe canalul de comunicaţie.

Astfel, prin crearea VLAN-urilor şi a domeniilor de *broadcast* separate, administratorii pot avea control asupra fiecărui port şi asupra fiecărui utilizator. Zilele când utilizatorii puteau să se conecteze când doreau cu staţia aflată la oricare din porturile unui switch, obţinând acces la resurse, ar trebui să fie istorie demult, deoarece administratorul are controlul asupra porturilor şi asupra resurselor ce se pot accesa prin acele porturi. Deşi VLAN-urile sunt create în concordantă cu resursele de care un utilizator are nevoie, switchurile pot fi configurate să informeze staţiile de management în legătură cu orice acces neautorizat sesizat. Iar dacă se doreşte şi comunicare între VLAN-uri se pot implementa restricţii pe router pentru a obţine un nivel înalt de securitate.

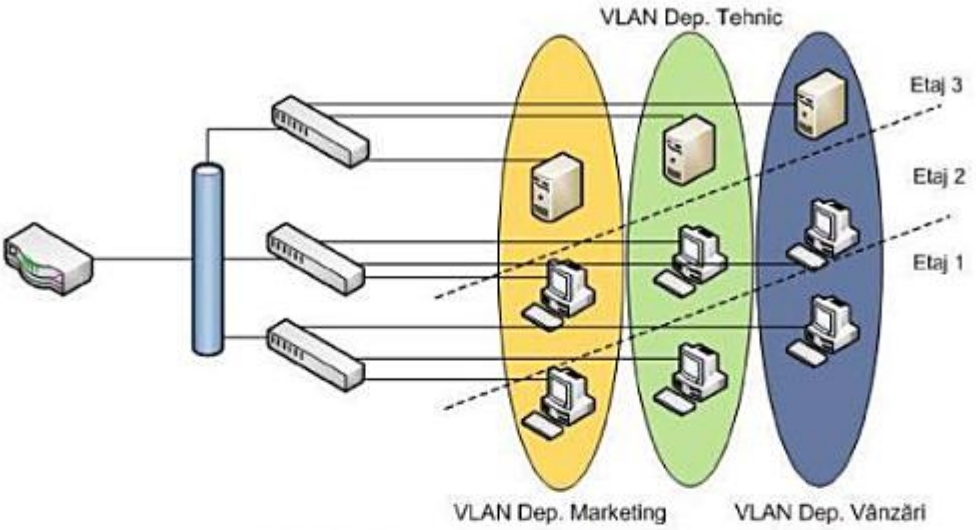
VLAN-urile oferă un grad mare de *flexibilitate şi scalabilitate*. Prin implementarea lor se creează domenii de *broadcast* mai mici. Asta înseamnă că *broadcast*-urile trimise de un nod dintr-un VLAN nu vor ajunge pe porturile configurate în alt VLAN. Aşa că, prin atribuirea porturilor sau utilizatorilor unui VLAN pe un switch, se câştigă flexibilitatea de a adăuga numai pe acei utilizatori pe care îi dorim în domeniul de *broadcast* respectiv. Această abordare poate funcţiona şi pentru blocarea *broadcast storm*-urilor cauzate de o placă de reţea defectă, cât şi pentru prevenirea unui echipament intermediar de a le propaga în toată reţeaua. Aceste *broadcast storm*-uri se pot întâmpla şi în cadrul unui VLAN, dar ele vor fi reduse numai la VLAN-ul din care au pornit.

VLAN-urile impun o mai mare *scalabilitate* într-o reţea comutată. Când un VLAN devine prea mare, se pot crea alte VLAN-uri, oprind astfel *broadcast*-urile să afecteze lăţimea de bandă a reţelei – cu cât sunt mai puţini utilizatori într-un VLAN, cu atât numărul *broadcast*-urilor este mai mic. Trebuie însă ţinut cont în mod special, când este vorba de proiectarea unei reţele cu VLAN-uri, de serviciile disponibile în reţea şi trebuie înţeles mecanismul conectării utilizatorilor la aceste resurse.

Un alt avantaj pe care îl oferă VLAN-urile este impactul sporit asupra performanţei reţelei. Putem grupa utilizatorii ce folosesc aplicaţii intensive de reţea într-un VLAN separat. De exemplu, putem crea un VLAN separat pentru un tehnician ce testează o aplicaţie multicast şi pentru serverele pe care acesta le foloseşte. Acesta se va bucura de timpi de răspuns foarte buni din partea resurselor pe care le utilizează, fiind într-un „LAN dedicat", în timp ce întregul departament tehnic nu îşi va mai exprima nemulţumirea în legătura cu scăderea ratei de transfer a filmului pe care aşteaptă să îl downloadeze, deoarece întreaga aplicaţie este izolată într-un VLAN separat.

VLAN-urile uşurează gestionarea migraţiilor, adăugărilor şi a schimbărilor din reţea – *network management*. Prin intermediul *software*-ului de pe switch se pot adăuga utilizatori într-un VLAN, iar mai târziu li se poate schimba apartenţa într-un altul. Recablarea pentru asigurarea conectivităţii nu mai este necesară într-o reţea comutată deoarece *tool*-urile de management ale reţelei permit reconfigurarea logică a sa în doar câteva secunde.

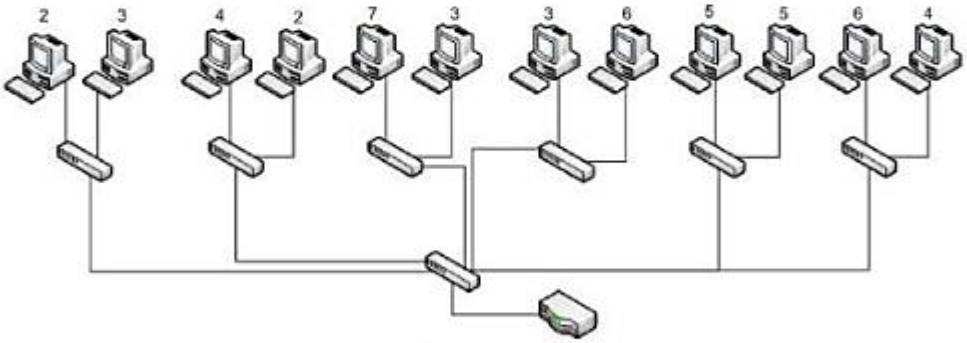
Material cu drept de autor



2-13: VLAN segmentat pe departamente

Figura de mai jos arată cum şase VLAN-uri (numerotate de la 2 la 7) au fost folosite pentru a crea un domeniu de *broadcast* pentru fiecare departament. Fiecărui port al switchului îi este atribuit câte un VLAN, depinzând de staţia şi domeniul de *broadcast* în care trebuie să fie aceasta.

Acum dacă se doreşte adăugarea unei noi staţii în VLAN-ul de Vânzări (VLAN 7), se atribuie portul respectiv VLAN-ului 7, indiferent de locaţia fizică a noului utilizator. Aceasta ilustrează unul dintre avantajele importante ale proiectării unei reţele cu VLAN-uri, şi anume reflectarea structurii organizatorice mai degrabă decât a structurii fizice.



2-14: Utilizarea VLAN-urilor

Marketing	VLAN2	172.16.2.0/24
Distribuţie	VLAN3	172.16.3.0/24
Tehnic	VLAN4	172.16.4.0/24
Finanţe	VLAN5	172.16.5.0/24
Management	VLAN6	172.16.6.0/24
Vânzări	VLAN7	172.16.7.0/24

Material cu drept de autor

De remarcat faptul că în figură VLAN-urile încep să fie numerotate de la VLAN 2. Oricum numărul este irelevant, dar ce s-a întâmplat cu VLAN 1? Acest VLAN este un VLAN administrativ, recomandându-se a fi folosit în acest scop. VLAN 1 nu poate fi şters sau modificat şi implicit, toate porturile de pe un switch fac parte din acest VLAN, până la o nouă atribuire.

Fiecare VLAN fiind considerat un domeniu de broadcast separat, trebuie să aibă un număr de rețea, prin care este identificat. Fiecare stație dintr-un VLAN nu poate comunica decât cu o stație din VLAN-ul respectiv, astfel că, pentru comunicația între VLAN-uri, este necesar un router sau un dispozitiv de nivel 3, aşa că nu e de aşteptat ca routerele să dispară prea curând!

Implementarea unui VLAN pe un switch implică următoarele acțiuni:

- switchul menține o tabelă de comutare separată pentru fiecare VLAN;
- dacă un cadru ajunge la switch pe portul ce se află într-un VLAN oarecare, switchul caută tabela de comutare pentru VLAN-ul respectiv;
- când cadrul este primit, switchul adaugă în tabela de comutare adresa sursă a cadrului recepționat dacă aceasta este necunoscută;
- switchul caută destinația pentru a şti ce decizie trebuie să ia;
- pentru partea de *learning* şi *forwarding*, căutarea este făcută în tabela de comutare a VLAN-ului respectiv.

2.3.1 Tipuri de VLAN-uri

VLAN-urile sunt de obicei create manual de către un administrator, ce atribuie apoi porturilor de pe switch câte un VLAN. Aceste VLAN-uri se numesc VLAN-uri **statice**. VLAN-urile statice sunt cele mai des folosite în ziua de azi, având numeroase avantaje cum ar fi: securitatea ridicată, uşurința de configurare, simplitatea monitorizării, funcționarea bună în rețele în care acțiunile sunt controlate şi administrate. Porturile îşi mențin configurațiile VLAN atribuite până când acestea sunt schimbate manual. Când o stație este conectată la un port al switchului, aceasta va face parte din VLAN-ul configurat pe portul respectiv.

În figura de mai sus, fiecare port al switchurilor a fost configurat manual cu câte un VLAN, în funcție de VLAN-ul în care fiecare stație trebuia să se afle – locația fizica a stației nu contează. De amintit faptul că fiecare stație trebuie să aibă o adresă logică (de nivel trei) consistentă. De exemplu, fiecare stație ce aparține VLAN-ului 2 trebuie configurată cu o adresă IP din rețeaua 172.16.2.0/24. De reținut faptul că înainte de a conecta o stație într-un port al unui switch, trebuie verificată configurația portului, pentru a verifica din ce VLAN face parte. Dacă VLAN-ul respectiv este diferit față de VLAN-ul în care trebuia stația să se afle, atunci stația va întâmpina probleme de accesare a resurselor locale VLAN-ului.

VLAN-urile **dinamice** determină apartenența unui nod la un VLAN automat, folosind managementul centralizat al aplicațiilor pe VLAN-uri. Deşi administrarea este mai simplă în centrele de cablare (*wiring closet*), VLAN-urile dinamice sunt mai rar folosite decât cele statice. Apartenența la un VLAN dinamic se bazează pe adresa fizică (MAC), adresa logică sau tipul de protocol.

De exemplu, se poate presupune că adresele fizice au fost introduse corect într-o bază de date, ce conține corespondențe <adresa MAC – VLAN> şi se află pe un server de configurat VLAN-uri. Dacă un nod este ataşat unui port de pe switch, neconfigurat în niciun VLAN, se va căuta în baza de date adresa MAC a nodului şi dacă aceasta va fi găsită, portul în care a fost conectat nodul va fi configurat şi i se va atribui VLAN-ul respectiv. Sistemul dispune şi de un mecanism de semnalizare când un utilizator necunoscut se adaugă la rețea (adresa MAC nu este configurată pe server).

Material cu drept de autor

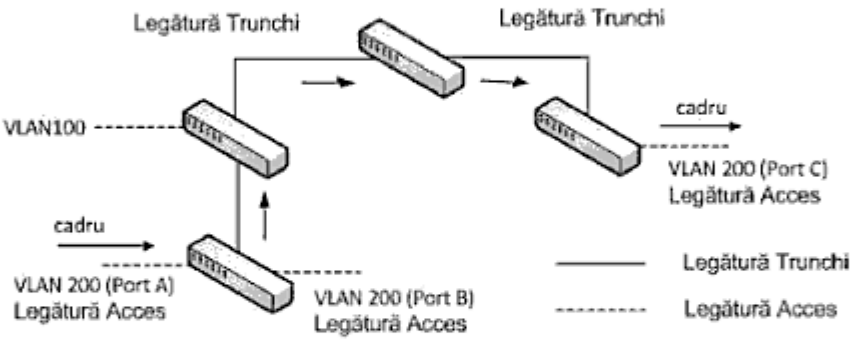
2.3.2 Legături acces/trunchi

Switchurile într-o reţea *Ethernet* trebuie să gestioneze toate tipurile de cadre şi, în plus, să înţeleagă ce să facă cu acestea, bazându-se pe adresa MAC. Tehnologia pusă la dispoziţie de VLAN-uri oferă posibilitatea grupării porturilor şi a utilizatorilor în grupuri logice, grupare ce implică folosirea mai multor switchuri, partajarea aceleiaşi clădiri sau a mai multor clădiri sau chiar WAN-uri. Pentru orice arhitectură VLAN, importantă este posibilitatea transferului de informaţie între staţii, switchuri şi routere.

Există două tipuri de legături într-o reţea bazată pe switchuri *Ethernet*: *legături de acces* şi *legături de trunchi*.

O **legătură de acces** reprezintă o legătură pe switchul ce este membru într-un singur VLAN şi este denumită „VLAN-ul nativ" al portului. Orice nod conectat printr-o legătură de acces nu este conştient de apartenenţa sa la vreun VLAN – nodul presupune că face parte dintr-un domeniu de broadcast, necunoscând aspectul fizic al reţelei.

Switchurile înlătură orice informaţie despre VLAN-uri, ce ar putea fi conţinută într-un cadru, înainte ca acesta să fie pus pe o legătură de acces. Staţiile de pe legăturile de acces nu pot comunica cu staţiile din alt VLAN, comunicarea realizându-se numai dacă pachetul este rutat.



2-15: Legături acces/trunchi

O **legătură de trunchi** este capabilă să suporte mai multe VLAN-uri, de obicei fiind folosită pentru a conecta switchurile de alte switchuri sau de routere. O legătură de trunchi poate fi privită ca o autostradă, pe care circulă maşini ce trebuie să ajungă fiecare la destinaţii diferite.

O legătură de trunchi nu aparţine unui VLAN specific, responsabilitatea acesteia fiind să acţioneze ca şi conexiune pentru VLAN-uri între switchuri şi routere. Poate fi configurată pentru a transporta toate VLAN-urile sau numai un număr limitat de VLAN-uri. Dacă legătura între switchuri nu este configurată drept legătură de trunchi, atunci implicit, numai informaţiile VLAN-ului 1 vor fi schimbate pe această legătură. O legătură de trunchi poate avea un VLAN nativ, acesta fiind folosit dacă legătură de trunchi cedează, dintr-un motiv oarecare.

2.3.3 Metode de identificare

După cum s-a observat mai sus, VLAN-urile pot fi create să se extindă pe o mulţime de switchuri, switchurile fiind conectate între ele prin legături de trunchi. Însă, problema care apare acum – chiar şi pentru un switch – este cum va gestiona acesta cadrele schimbate pe legătura de trunchi şi cum identifică VLAN-ul de care aparţine un cadru?

Conceptul ce a fost introdus se bazează pe asignarea de către switch a unui identificator unic fiecărui cadru, identificator ce reprezintă VLAN-ul din care face parte (*VLAN ID*). Această metodă de modificare a cadrelor primite de switch poartă numele de *frame tagging*. Fiecare

Material cu drept de autor

switch ce primeşte un cadru, conţinând un marcaj de VLAN (*tag*), trebuie mai întâi să identifice identificatorul de VLAN (VLAN ID), după care switchul se uită în tabela de filtrare pentru a lua o decizie. Dacă switchul ce a primit cadrul are numai o legătură de trunchi, atunci cadrul va fi trimis pe aceasta. Însă, o dată ce un cadru trebuie să ajungă pe o legătură de acces, se verifică identificatorul de VLAN cu VLAN-ul ce se află pe acea legătură, iar dacă acestea corespund, switchul înlătură VLAN ID-ul şi trimite cadrul pe acel segment de reţea.

Scopul primar al metodelor de trunchiere este acela de a asigura comunicaţia între switchuri, într-o arhitectură bazată pe VLAN-uri. Cisco a creat ISL, acesta funcţionând numai între echipamente Cisco. Dacă se doreşte un protocol de trunchiere ce nu este proprietar, se va folosi IEEE 802.1Q. În cele ce urmează, va fi analizat standardul 802.1Q, metoda de trunchiere ISL ne mai fiind suportată pe nicio platformă Cisco Catalyst.

802.1Q reprezintă standardul IEEE pentru marcarea cadrelor (*frame tagging*) ce traversează o legătură de trunchi. Fiind un standard IEEE, acest protocol poate fi folosit cu uşurinţă între echipamentele ce provin de la diferiţi producători. 802.1Q nu încapsulează cadrul original, ci în schimb, inserează un câmp de 4 octeţi în antetul cadrului *Ethernet* original, recalculând suma de control (FCS), înainte ca acesta să fie trimis pe o legătură de trunchi. Overheadul pe care acest protocol îl introduce este de 4 octeţi, ceea ce duce la o lungime maximă a cadrului *Ethernet* de 1522 de octeţi.

Câmpul de 4 octeţi, inserat între câmpurile Adresă Sursă şi Lungime/Tip, este format din două părţi: primii 2 octeţi indică identificatorul protocolului VLAN, ce are întotdeauna valoarea 0x8100, iar următorii 2 octeţi conţin trei câmpuri. Sub-câmpul cel mai important este identificatorul de VLAN, ce ocupă cei mai puţin semnificativi 12 biţi, ceea ce indică faptul că o legătură de trunchi ce utilizează protocolul 802.1Q poate suporta până la 4096 de VLAN-uri.

Câmpul Prioritate (*Priority*) de 3 biţi se referă la standardul IEEE 802.1p, de prioritate a traficului. Acest câmp face distincţia între traficul în timp real implementat hard şi cel implementat soft şi de traficul intens pentru o mai bună calitate a serviciilor în *Ethernet*.

1	6	6	2	46-1500	4
Preambul	Adresă destinaţie	Adresă sursă	Lungime/Tip	Date	Sumă de control

2-16: Structura cadrului Ethernet 802.3

1	6	6	4	2	46-1500	4
Preambul	Adresă destinaţie	Adresă sursă	TAG	Lungime/Tip	Date	Sumă de control

2-17: Structura cadrului 802.1Q

2B	3b	1b	12b
Identificator protocol (0x8100)	Prioritate	CFI	VLAN ID

2-18: Structura câmpului TAG

Câmpul CFI (*Canonical Format Indicator*) de 1 bit era folosit să indice adresele MAC în format *Little Endian* sau *Big Endian*. În prezent este un *flag* pentru anunţarea existenţei unui cadru prestabilit 802.5 (Token Ring).

Notă: Ce trebuie remarcat este faptul că această metodă de trunchiere introduce o supraîncărcare de 4 octeţi cadrului *Ethernet*. Deoarece cadrele *Ethernet* nu trebuie să fie mai mari de 1518 octeţi, informaţia adiţională ce este adăugată va crea cadre ce depăşesc

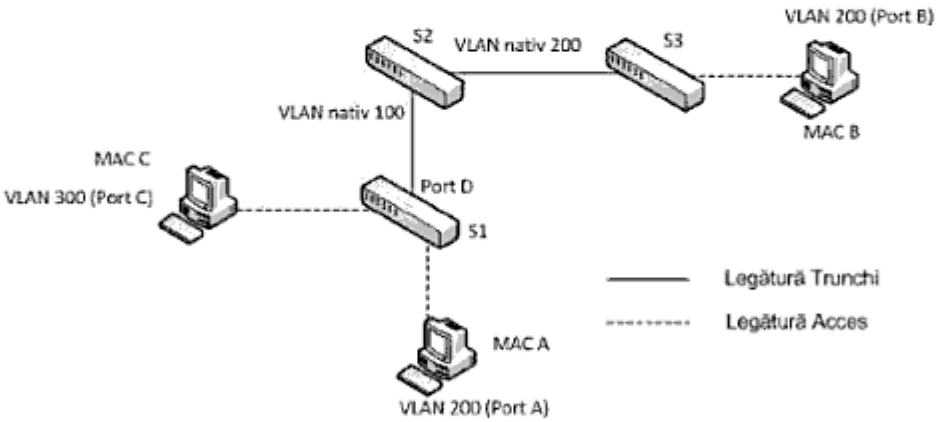
Material cu drept de autor

lungimea maximă admisibilă. Acestea sunt denumite cadre *„baby giant”*, switchurile raportând astfel de cadre drept erori sau cadre prea mari. Pentru a rezolva această problemă, switchurile trebuie să înţeleagă standardul IEEE 802.3ac, ce extinde lungimea maximă a cadrului *Ethernet* la 1522 octeţi.

Standardul 802.1Q introduce conceptul de VLAN nativ. Cadrele ce aparţin acestui VLAN nu sunt modificate când sunt transportate pe o legătură de trunchi. VLAN-urile native mai sunt cunoscute sub numele de VLAN-uri de management. Când se configurează 802.1Q pe o legătură de trunchi, trebuie configurat acelaşi VLAN nativ în ambele părţi ale trunchiului. Implicit, VLAN-ul nativ este VLAN 1, iar toate porturile sunt în acest VLAN când switchul este pornit.

VLAN-ul nativ este utilizat pentru tot traficul nemarcat primit pe o legătură trunchi, configurată cu 802.1Q. Această opţiune este dorită în special numai pentru a asigura comunicaţia directă, fără modificarea cadrelor, între porturile capabile de 802.1Q şi cele mai vechi 802.3. Totuşi, în toate celelalte cazuri, poate deveni foarte dăunătoare (*VLAN hopping*), deoarece cadrele asociate cu VLAN-ul nativ îşi pierd nu numai marcajul, identificatorul, ci şi clasa de prioritate (biţii de prioritate) când sunt transportate pe o legătură 802.1Q. Din aceste motive – pierderea mijloacelor de identificare şi de clasificare – folosirea VLAN-ului nativ ar trebui evitată. În cazurile în care acest lucru nu poate fi făcut, întotdeauna se alege un VLAN nefolosit, diferit de cel implicit, pentru VLAN-ul nativ al tuturor legăturilor trunchi.

Pentru a exemplifica lucrurile menţionate până acum, se consideră figura de mai jos. Staţia A ce se află conectată pe portul A al switchului 1 (S1), doreşte să trimită un pachet staţiei B, ce se află conectată pe portul B al switchului 3 (S3). Legăturile de trunchi au VLAN-urile native configurate astfel: legătura dintre S1 şi S2 este configurată cu VLAN nativ 100, iar legătura dintre S2 şi S3 este configurată cu VLAN nativ 200. Ambele staţii de află în VLAN 200.



2-19: Folosirea VLAN-ului nativ

Înainte de descrierea efectivă a cadrelor ce sunt trimise între cele două staţii, cum este organizată tabela de comutare a switchului S1? Aceasta va fi organizată pe secţiuni separate pentru fiecare VLAN. Cu alte cuvinte, fiecare asociere din tabele de comutare, pe lângă perechea <adresa_MAC-port>, va avea specificat şi VLAN-ul căruia îi aparţine.

VLAN	Interfaţă/port	Adresă MAC
200	A	MAC A
300	C	MAC C
200	D	MAC B

2-20: Tabela de comutare a switchului S1

Material cu drept de autor

După ce stația A află toate informațiile necesare pentru construirea pachetului de date, îl transmite, acesta ajungând pe portul A al switchului S1. Switchul S1 citește adresa MAC destinație și verifică tabela de comutare pentru o eventuală asociere. Așa cum se observă și în tabelul de mai sus, pachetul va trimis pe portul D al switchului S1. Dar, înainte de trimiterea pachetului, S1 va trebui să adauge informația de nivel doi (VLANID), pe baza căreia switchul S2 va ști cărui VLAN să trimită cadrul. Se observă, ca dacă VLAN-ul nativ este diferit de VLAN-ul din care face parte stația ce transmite, switchul ce primește cadrul va trebui sa adauge identificatorul de VLAN pentru pachetul respectiv. Când pachetul este primit de switchul S2, acesta va urma aceeași pași de decizie, în schimb, va observa că VLAN-ul nativ al legăturii de trunchi pe care trebuie să transmită cadrul este același cu cel al stației ce a transmis cadrul respectiv (stația A). În urma acestui lucru, identificatorul de VLAN este șters și pachetul este trimis switchului S3, ce îl va trimite mai departe stației B.

Succesiunea de pachete este următoarea:

De la stația A la switchul S1:

MAC destinație	MAC sursă	Lungime/Tip	Date	CRC
MAC B	MAC A	0x0800	X	X

De la switchul S1 la switchul S2:

MAC B	MAC A	0x8100	200	0x0800	X	X

De la switchul S2 la switchul S3:

MAC B	MAC A	0x0800	X	X

De la switchul S3 la stația B:

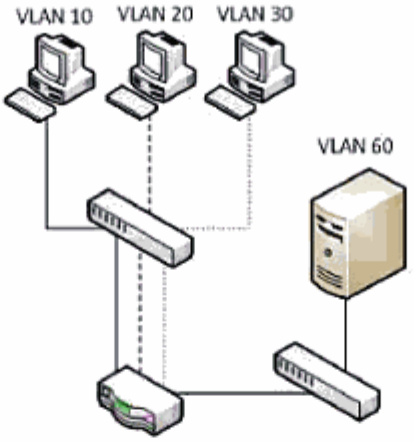
MAC B	MAC A	0x0800	X	X

2.4 *Rutare între VLAN-uri*

Stațiile dintr-un VLAN fac parte din domeniul de *broadcast* definit de VLAN-ul respectiv, putând comunica între ele. VLAN-urile partiționează rețeaua și separă traficul de nivel doi, astfel încât dacă dorim ca două stații din VLAN-uri diferite să comunice între ele este nevoie de un dispozitiv de nivel trei și anume un router. Deoarece pentru fiecare VLAN se utilizează de obicei o adresă de rețea, comunicația între VLAN-uri ar fi imposibilă fără utilizarea unui router. Pentru aceasta se poate folosi un router cu o interfață pentru fiecare VLAN, metodă însă foarte costisitoare pentru un număr mare de VLAN-uri și rar folosită, sau un router cu o interfață care suportă 802.1Q, interfață ce va fi conectată la un port de *trunking*.

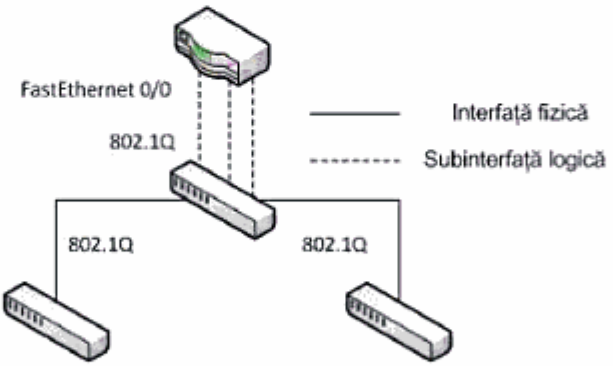
Pentru un număr mic de VLAN-uri (două sau maxim trei) se poate folosi un router cu două sau trei interfețe *Ethernet* (Fast *Ethernet*), așa cum se arată în figura de mai jos. Fiecare legătură a switchului cu routerul reprezintă o legătură de acces, rutarea între VLAN-uri fiind o rutare clasică, fiecare VLAN fiind văzut ca o rețea separată.

Material cu drept de autor



2-21: Router conectând patru VLAN-uri, folosind interfeţe dedicate

Însă, dacă numărul VLAN-urilor creşte, această metodă este cu adevărat costisitoare din punct de vedere al costului routerului. În schimb, putem configura o interfaţă *Ethernet* sau Fast *Ethernet* a routerului pentru suport 802.1Q. În loc să se folosească interfeţe pentru fiecare VLAN, se poate utiliza doar o singură interfaţă a routerului ce va crea o legătură de trunchi între router şi switch, routerul numindu-se în acest caz "*router-on-a-stick*", aşa cum se arată în figura de mai jos. Interfaţa de trunchi va fi împărţită în subinterfeţe, fiecărei subinterfeţe atribuindu-i-se o adresă IP şi o metodă de trunchiere.



2-22: Router ce asigură comunicaţia între VLAN-uri, folosind doar o singură interfaţă fizică (router on a stick).

Notă: *Majoritatea routerelor nu suportă trunking pe interfeţele Ethernet, deşi actualmente există IOS (Internetwork Operating System) pentru modelele 2610 ce oferă acest lucru.*

2.5 Rezumat

După cum s-a prezentat, *Ethernet* se bazează pe un mediu de tip partajat (*shared-media*), deci numai o singură staţie poate transmite la un moment dat. Astfel, dacă numărul nodurilor pe un segment creşte, probabilitatea de apariţie a coliziunilor se va mări. Domeniul de coliziune este acea zonă dintr-o reţea care va fi afectată de apariţia unei coliziuni în interiorul ei. Reţeaua locală poate fi împărţită în domenii de coliziune separate prin intermediul unor dispozitive din categoria switchurilor.

Material cu drept de autor

Switchurile construiesc dinamic şi menţin o tabelă de asocieri între adresele MAC şi una din interfeţele sale, numită tabelă de comutare. Această tabelă este construită pe baza adresei sursă a cadrului primit pe unul dintre porturile switchului, în tabela de comutare fiind introdusă asocierea <MAC_sursă – port_intrare>.

Impactul redundanţei asupra reţelelor formate numai din switchuri este esenţial. Pentru identificarea buclelor de nivel legătură de date şi întreruperea lor s-a dezvoltat protocolul numit STP - *Spanning Tree Protocol*, ce porneşte de la construirea unui arbore de acoperire pe graful determinat de dispozitivele de interconectare şi de conexiunile dintre acestea.

O dezvoltare în domeniul interconectării LAN-urilor este posibilitatea de separare a topologiei logice de topologia fizică, lucru realizat prin VLAN-uri (*Virtul LAN*). VLAN-urile împart o reţea bazată pe switchuri în domenii de *broadcast* separate, un lucru foarte important şi necesar din cauză că switchurile de nivel doi împart reţeaua în domenii de colizune independente, neavând niciun efect asupra domeniilor de *broadcast*. Un nou format pentru cadrele *Ethernet* (802.1Q) a fost introdus pentru a oferi o modalitate mai simplă de introducere a VLAN-urilor în organizaţii.

2.6 Studiu de caz:

2.6.1 Comenzi pe switchuri Cisco

În acest studiu de caz se vor prezenta câteva *output*-uri de comenzi de pe switchurile Catalyst model 2950 .

După cum se ştie, switchurile învaţă dinamic adresele staţiilor conectate la porturile lor. În figura de mai jos switchul A are trei astfel de adrese în tabela de comutare, staţiile fiind conectate pe porturile 14, 16 şi 17. Se observă patru intrări statice în tabela de comutare. Prima adresă MAC este cea a switchului, adresă ce face parte din spaţiul de adrese fizice gestionat de Cisco[1], iar celelalte trei sunt adrese MAC virtuale folosite de CatOS (*Catalyst Operating System*) pentru adresarea *multicast*.

```
SwitchA#show mac-address-table
          Mac Address Table
-------------------------------------------
Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
 All    0008.219f.5e40    STATIC      CPU
 All    0100.0ccc.cccc    STATIC      CPU
 All    0100.0ccc.cccd    STATIC      CPU
 All    0100.0cdd.dddd    STATIC      CPU
   1    0004.4dbb.f220    DYNAMIC     Fa0/14
   1    0004.9a9d.56a0    DYNAMIC     Fa0/16
   1    0008.a326.13c4    DYNAMIC     Fa0/17
Total Mac Addresses for this criterion: 7
```

Pentru afişarea doar a intrărilor dinamice se foloseşte comanda:

```
SwitchA#show mac-address-table dynamic
          Mac Address Table
-------------------------------------------
Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    0004.4dbb.f220    DYNAMIC     Fa0/14
   1    0004.9a9d.56a0    DYNAMIC     Fa0/16
   1    0008.a326.13c4    DYNAMIC     Fa0/17
Total Mac Addresses for this criterion: 3
```

Fiecare adresă învăţată de switch este ţinută în tabela de comutare o anumită perioadă de timp numită timp de îmbătrânire (*aging time*), valoare implicită fiind de 300 de secunde.

```
SwitchB#show spanning-tree
VLAN0001
  Spanning tree enabled protocol ieee
  Root ID     Priority    32768
```

[1] http://standards.ieee.org/regauth/oui/oui.txt

Material cu drept de autor

```
                    Address     0008.a327.8900
                    Cost          19
                    Port          6 (FastEthernet0/6)
                    Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec

  Bridge ID  Priority         32769  (priority 32768 sys-id-ext 1)
                Address       0008.219f.5e40
                Hello Time    2 sec  Max Age 20 sec  Forward Delay 15 sec
                Aging Time    300

Interface        Role Sts Cost      Prio.Nbr Type
---------------- ---- --- --------- -------- --------------------------------
Fa0/4            Desg FWD 19        128.4    P2p
Fa0/6            Root FWD 19        128.6    P2p
Fa0/9            Desg FWD 19        128.9    P2p
Fa0/12           Desg FWD 19        128.12   P2p
```

Mai sus este prezentată ieşirea comenzii *show spanning-tree*. Această comandă este utilă când se doreşte aflarea rădăcinii din propria reţea, prioritatea şi adresa MAC a acestuia. Totodată se observă costul portului prin care se conectează *root bridge*-ul la switch. Acesta are valoarea de 19, ceea ce indică faptul că switchul este conectat direct la *root bridge* pe portul 6. Valorile timpilor de trimite a cadrelor BPDU (default 2 sec.) şi de trecere din starea de *blocking* în *listening* (*Max Age*) şi apoi în *learning* şi *forwarding* (*Forward Delay*) sunt la valorile implicite. Se observă aici rolul (port rădăcină – Fa0/6) şi statusul fiecărui port de pe switch, toate cele patru fiind în starea de *forwarding*.

```
SwitchA#show mac-address-table aging-time
Vlan    Aging Time
----    ----------
300
```

Dacă se doreşte numai o scurtă prezentarea a opţiunilor STP se poate folosi comanda `show spanning-tree summary`, ceea ce indică numărul porturilor ce se află în diferitele stări STP şi VLAN-ul din care acestea fac parte. În ieşirea de mai jos patru porturi sunt în starea de forwarding, în VLAN-ul 1.

```
SwitchB#show spanning-tree summary
Switch is in pvst mode
Root bridge for: none
EtherChannel misconfig guard          is enabled
Extended system ID                    is enabled
Portfast Default                      is disabled
PortFast BPDU Guard Default           is disabled
Portfast BPDU Filter Default          is disabled
Loopguard Default                     is disabled
UplinkFast                            is disabled
BackboneFast                          is disabled
Pathcost method used                  is short

Name                   Blocking Listening Learning Forwarding STP Active
---------------------- -------- --------- -------- ---------- ----------
VLAN0001                    0         0         0          4          4
---------------------- -------- --------- -------- ---------- ----------
1 vlan                      0         0         0          4          4
```

Dacă se doreşte vizualizarea recalculării topologiei de către STP şi trecerea porturilor prin diferitele stări la schimbarea unei staţii de pe un port al switchului pe altul, se poate folosi comanda:

```
SwitchB#debug spanning-tree events
Spanning Tree event debugging is on
SwitchB #
01:06:53: STP: VLAN0001 we are the spanning tree root
SwitchB #
01:06:54: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/6, chann
SwitchB #
01:06:57: set portid: VLAN0001 Fa0/6: new port id 8006
01:06:57: STP: VLAN0001 Fa0/6 -> listening
SwitchB #
01:06:58: %LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/6, chanp
SwitchB #
01:07:02: STP: VLAN0001 heard root 32768-0008.a327.8900 on Fa0/6
01:07:02:      supersedes 32769-0008.219f.5e40
01:07:02: STP: VLAN0001 new root is 32768, 0008.a327.8900 on port Fa0/6, cost 19
01:07:02: STP: VLAN0001 sent Topology Change Notice on Fa0/6
SwitchB #
```

Material cu drept de autor

```
01:07:12: STP: VLAN0001 Fa0/6 -> learning
SwitchB #
01:07:27: STP: VLAN0001 sent Topology Change Notice on Fa0/6
01:07:27: STP: VLAN0001 Fa0/6 -> forwarding
```

La pornirea switchului toate porturile se află în VLAN-ul 1 până la o nouă atribuire. VLAN-urile 1002 şi 1004 sunt rezervate pentru reţelele FDDI (*Fiber-Distributed Data Interface*), în timp ce VLAN-urile 1003 şi 1005 pentru reţelele *Token Ring*.

```
SwitchC#show vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0/1, Fa1/0/2, Fa1/0/3
                                                Fa1/0/4, Fa1/0/5, Fa1/0/6
                                                Fa1/0/7, Fa1/0/8, Fa1/0/9
                                                Fa1/0/10, Fa1/0/11, Fa1/0/12
                                                Fa1/0/13, Fa1/0/14, Fa1/0/15
                                                Fa1/0/16, Fa1/0/17, Fa1/0/18
                                                Fa1/0/19, Fa1/0/20, Fa1/0/21
                                                Fa1/0/22, Fa1/0/23, Fa1/0/24
                                                Gi1/0/1, Gi1/0/2, Gi1/1/1
1002 fddi-default                     act/unsup
1003 trcrf-default                    act/unsup
1004 fddinet-default                  act/unsup
1005 trbrf-default                    act/unsup
```

După crearea VLAN-urilor şi a legăturilor de acces se observă adăugarea acestora în tabelă:

```
SwitchC(config)#interface FastEthernet 1/0/14
SwitchC(config-if)#switchport mode access
SwitchC (config-if)#switchport access vlan 2
SwitchC #show vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa1/0/1, Fa1/0/2, Fa1/0/3
                                                Fa1/0/4, Fa1/0/5, Fa1/0/6
                                                Fa1/0/7, Fa1/0/8, Fa1/0/9
                                                Fa1/0/10, Fa1/0/11, Fa1/0/12
                                                Fa1/0/13, Fa1/0/15, Fa1/0/17
                                                Fa1/0/18, Fa1/0/19, Fa1/0/20
                                                Fa1/0/21, Fa1/0/22, Fa1/0/23
                                                Fa1/0/24, Gi1/0/1, Gi1/0/2
                                                Gi1/1/1, Gi1/1/2
2    VLAN0002                         active    Fa1/0/14
3    VLAN0003                         active    Fa1/0/16
```

Notă: *Dacă se atribuie un port unui VLAN ce nu există, portul este inactiv până ce se creează VLAN-ul respectiv.*

Numai porturile configurate ca porturi de acces sunt afişate.

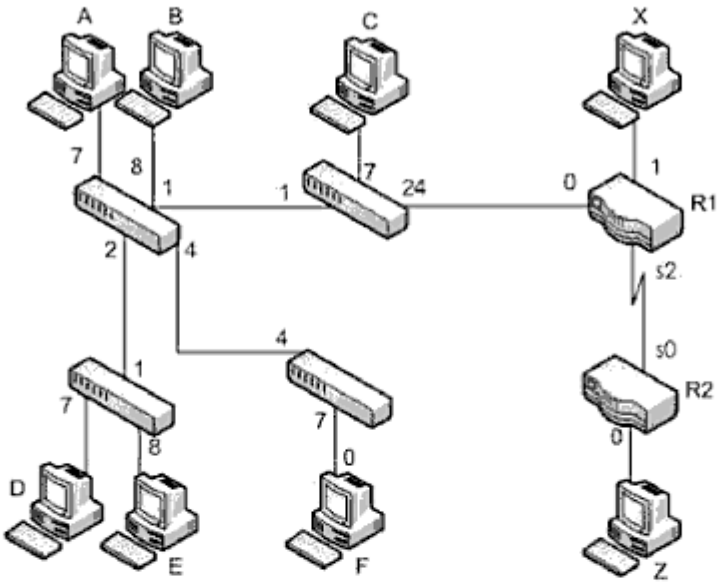
```
SwitchC (config)#int fastEthernet 1/0/6
SwitchC (config-if)#switchport mode trunk
SwitchC (config-if)#switchport trunk  encapsulation dot1q

SwitchC #show interface fastEthernet 1/0/6 switchport
Name: Fa1/0/6
Switchport: Enabled
Administrative Mode: dynamic auto
Operational Mode: down
Administrative Trunking Encapsulation: dot1q
Negotiation of Trunking: On
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 1 (default)
<...>
```

După crearea legăturilor de trunchi, interfeţele pe care s-au configurat metodele de trunchiere vor fi marcate. Această comandă afişează: numele portului, modul administrativ şi operaţional al portului, metoda de încapsulare, modul de negociere, toate opţiunile fiind în starea implicită, cu excepţia metodei de trunchiere. Modul implicit pentru porturile unui switch este *dynamic auto*. Dacă interfaţa vecină cu care se conectează un port de pe switch suportă trunchiere şi este configurată în modul de trunchiere, legătura dintre cele două porturi devine una de trunchi. Implicit, legăturile de trunchi negociază metodele de

Material cu drept de autor

încapsulare, metodă gestionată de DTP – *Dynamic Trunking Protocol*. Dacă interfața vecină suportă ISL și 802.1Q și ambele interfețe sunt configurate să negocieze metoda de încapsulare, pe legătura de trunchi se va folosi ISL.

2.6.2 Încapsularea pachetelor: dot1q



Pentru topologia din imagine se definesc VLAN-uri astfel:

Toate stațiile conectate pe port impar vor fi în `VLAN 100`, cele pe port par în `VLAN 200`

Legătura `Sw1-Sw4` va fi trunchi cu `VLAN nativ 100`, restul legăturilor vor fi configurate ca și legături de trunchi cu `VLAN nativ 200`

1. Scrieți toate antetele cadrelor ce vor apare în cazul în care stația `C` îi trimite un singur cadru stației `F`.
2. Câte domenii de coliziune și câte domenii de *broadcast* sunt în toată topologia?
3. Toată rețeaua a fost reinițializată. Sunt trimise 3 pachete în rețea: Stația `A` trimite un pachet către stația `C`, apoi un pachet către `D`. F trimite un pachet către `A`. Ce intrări vor fi în tabela de comutare a switchului `Sw2`?

Rezolvare:

1. Succesiunea de pachete este următoarea:

De la stația `C` la switchul `Sw2`:

MAC destinație	MAC sursă	Lungime/Tip	Date	CRC
MAC F	MAC C	0x0800	X	X

De la switchul `Sw2` la switchul `Sw1`:

MAC F	MAC C	0x8100	100	0x0800	X	X

De la switchul `Sw1` la switchul `Sw4`:

MAC F	MAC C	0x0800	X	X

De la switchul `Sw4` la stația `F`:

MAC F	MAC C	0x0800	X	X

Material cu drept de autor

2a. Domeniile de coliziune sunt mărginite de switchuri şi routere. Fiecare port al switchului se află în alt domeniu de coliziune. Pentru legăturile *full-duplex* nu vor mai exista coliziuni, astfel porturile switchului negociate în *full-duplex* nu vor fi numărate ca domenii distincte de coliziune. În cazul problemei de față nu sunt precizate porturile ce ajung în starea *half-duplex*, astfel se vor considera cele două cazuri extreme: toate interfețele din rețea sunt *half-duplex* şi cazul al doilea în care toate interfețele sunt *full-duplex*.

Dacă toate porturile sunt *half-duplex* în rețea vor exista 12 domenii de coliziune:

`{Sw4(7), E}; {Sw3(7), D}; {Sw3(8), E}; {Sw1(7), A}; {Sw1(8), B}; {Sw2(7), C}; {Sw1(1), Sw2(1)}; {Sw1(2), Sw3(1)}; {Sw1(4), Sw4(4)}; {R1(e0), Sw2(24)}; {R1(e1), X}; {R2(e1), Z}.`

Rețeaua dintre routerele `R1` şi `R2` este o rețea punct-la-punct în care nu pot exista coliziuni.

Pentru cel de al doilea caz în care toate legăturile sunt *full-duplex* nu va exista niciun domeniu de coliziune în rețea.

2b. Domeniile de difuzare sunt conectate doar de routere. Prin definirea de VLAN-uri şi switchurile limitează domenii de *broadcast*. În cazul topologiei singura modalitate de a asigura conectivitatea între `VLAN 100` şi `VLAN 200` este configurarea routerului `R1` ca *router-on-a-stick*. Pentru aceasta legătura dintre `Sw2` şi `R1` trebuie să transporte ambele VLAN-uri.

Problematica domeniilor de *broadcast* se aplică doar rețelelor multiacces, astfel pentru rețeaua dintre `R1` şi `R2` (legătura serială) nu va exista un domeniu de difuzare.

Cele 4 domenii de broadcast sunt:

`{R1(e0), A, C, D, F, Sw1(1), Sw1(2), Sw1(4), Sw1(7), Sw2(1), Sw2(7), Sw2(24), Sw3(1), Sw3(7), Sw4(4), Sw4(7)}`

`{R1(e0), B, E, Sw1(1), Sw1(2), Sw1(4), Sw1(8), Sw2(1), Sw2(24), Sw3(1), Sw3(8), Sw4(4)}`

`{R1(e1), X}`

`{R2(e0, Z}`

3. Deoarece rețeaua a fost reinițializată, tabelele de comutare ale switchurilor nu au nicio intrare. Când stația `A` va trimite un pachet către stația `C`, switchul `Sw1`, având tabela de comutare goală, va trimite pachetul pe toate porturile mai puțin cel pe care a sosit. După cum se ştie switchurile îşi populează tabelele de comutare pe baza adreselor `MAC` sursă a cadrelor primite. Astfel, switchul `Sw1`, înainte de trimiterea pachetului primit de la stația `A`, va scrie în tabela de comutare asocierea <`MAC A - port 7`>. La fel, switchul `Sw4` va scrie în tabela de comutare asocierea <`MAC A - port 4`>. Când pachetul va ajunge la switchul `Sw2`, acesta va proceda similar, trimițându-l pe toate porturile mai puțin cel pe care a venit, populând totodată tabela de comutare cu asocierea <`MAC A - port 1`>.

Când stația `A` va trimite un pachet către stația `D`, procesul de învățare şi de trimitere este similar. În acest caz, switchul `Sw2` nu va mai scrie în tabela de comutare asocierea `MAC sursă - port` (<`MAC A - port 1`>), această intrare existând deja.

Când stația `F` va trimite un pachet către stația `A`, switchul `Sw4` îl va primi, verificând adresa destinație, pe baza căreia va lua o decizie. Cum adresa `MAC` destinație este cea a lui `A` (`MAC A`) şi cum în tabela de comutare există o intrare de tipul <`MAC A - port 4`>, switchul `Sw4` va trimite pachetul pe portul `4`, acesta ajungând la switchul `Sw1`. Switchul `Sw1` va verifica tabela de comutare şi va descoperi intrarea <`MAC A - port 7`>, ceea ce îl determină să trimită pachetul pe portul `7`, unde există stația `A`.

Astfel, tabela de comutare a switchului `Sw2` va avea în urma trimiterii celor 3 pachete o singură intrare şi anume <`MAC A - port 1`>.

Adresă MAC	Port
MAC A	1

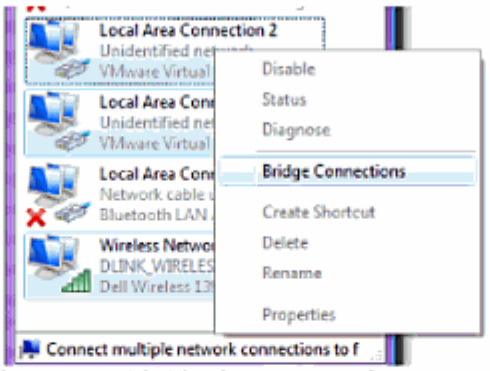
2-23: Tabela de comutare a switchului Sw2

Material cu drept de autor

2.7 *Realizarea unui bridge între conexiuni în* Windows Server 2008

În unele cazuri şi topologii este de dorit să se combine mai multe conexiuni de reţea de pe acelaşi calculator astfel încât Windows să le trateze ca pe singură reţea iar membrii reţelelor să poată comunica între ei perfect transparent. De asemenea, această tehnică presupune şi includerea tuturor reţelelor aflate în *bridging* într-un singur domeniu de *broadcast*.

Un alt avantaj al *bridging*-ului este faptul că pot fi combinate reţele de tehnologii diferite (*wireless*, *Ethernet*, chiar şi *Token Ring*), atâta timp cât echipamentul care realizează *bridging*-ul deţine interfeţele corespunzătoare în fiecare reţea.



2-24: Realizarea unui *bridge* între mai multe conexiuni de reţea

Tehnica de realizare a unui *brigde* între mai multe conexiuni de reţea, pe *Windows Server 2008* se reduce la selectarea a două sau a mai multor conexiuni de reţea din *Network Connections* şi la alegerea opţiunii de *Bridge Connections* din meniul contextual obţinut prin clic dreapta.

Pe un calculator poate fi definit un singur *bridge*, dar acesta poate conţine oricâte reţele. *Bridge*-ul poate fi eliminat în orice moment din interfata *Network Connections* printr-un clic dreapta pe el si alegerea opţiunii de ştergere. În mod asemănător, o conexiune poate fi eliminată dintr-un *bridge*.

Atenţie! *Realizarea unui bridge între o conexiune spre o reţea locală şi una spre Internet este posibilă dar trebuie ţinut cont de faptul că toate staţiile din reţeaua locală vor trebui să poată avea acces la Internet şi să primească adrese din acelaşi subnet ca şi adresa externă a gateway-ului, ceea ce nu e întotdeauna posibil din partea ISP-ului. De asemenea, trebuie avut în vedere şi faptul că expunerea staţiilor din reţeaua locală în Internet poate reprezenta un risc de securitate.*

Material cu drept de autor

Întrebări

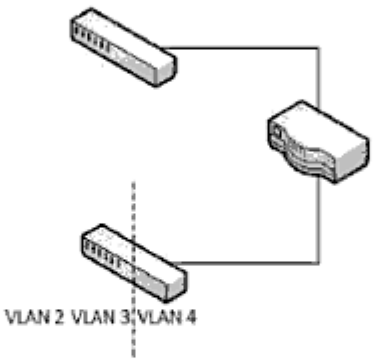
1. *Care este rezultatul segmentării unei reţele cu un switch ?*

- *Creşte numărul domeniilor de coliziune*
- *Scade numărul domeniilor de coliziune*
- *Creşte numărul domeniilor de broadcast*
- *Scade numărul domeniilor de broadcast*

2. *Care este numărul minim de adrese MAC asociate cu un switch de nivel doi?*

- *Una*
- *Două*
- *Atâtea câte porturi există*
- *Niciuna*

3. *Câte domenii de broadcast sunt în figura de mai jos ?*



- *1*
- *2*
- *3*
- *4*

4. *Ce nivel din stiva OSI este folosit de către switchurile Ethernet pentru a lua o decizie ?*

- *Nivelul 1*
- *Nivelul 2*
- *Nivelul 3*
- *Nivelul 4*

5. *Pe baza cărei informaţii un switch Ethernet poate lua o decizie ?*

- *adresa IP*
- *adresa MAC destinaţie*
- *adresa CAM*
- *adresa MAC sursă*

6. *Ce metodă de comutare citeşte primii 64 de octeţi ai cadrului, înainte de trimitearea acestuia?*

Material cu drept de autor

- *fast-forward*
- *cut-through*
- *fragment-free*
- *store and forward*

7. *Care dintre următoarele afirmaţii este adevărată despre VLAN-uri?*
- *Trebuie să existe cel puţin două VLAN-uri definite într-o reţea*
- *Toate VLAN-urile sunt configurate pe porturile cele mai rapide, şi implicit, informaţia se transmite celorlalte switchuri din reţea*
- *Reduc dimensiunea domeniului de broadcast*
- *VLAN-urile măresc numărul de switchuri dintr-o reţea*

8. *Care dintre următoarele afirmaţii este adevărată cu privinţă la comutarea de nivel doi (alegeţi două variante)?*
- *Un switch este un hub cu mai multe porturi*
- *Un switch este o punte cu mai multe porturi*
- *Switchurile învaţă adresele IP ale cadrelor şi iau decizii pe baza acestora*
- *Switchurile învaţă adresele MAC, examinând câmpul sursă al fiecărui cadru*

9. *Switchul dumneavoastră trebuie să fie setat ca root bridge. Care dintre următoarele posibilităţi va face acest switch să devină root bridge?*
- *Setarea adresei MAC a switchului la o valoare minimă*
- *Setarea protocolului STP la o valoare minimă*
- *Setarea priorităţii switchului la o valoare maximă*
- *Setarea priorităţii switchului la o valoare minimă*

10. *Un switch Ethernet dispune de o tabelă de comutare ca cea din figura de mai jos. Ce decizie va lua switchul dacă va primi cadrul cu adresa destinaţie 00-00-3D-1F-11-03 şi adresa sursă 00-00-3D-1F-11-01:*

Staţia	Port 1	Port 2	Port 3	Port 4
00-00-3D-1F-11-01			X	
00-00-3D-1F-11-02				
00-00-3D-1F-11-03	X			X

- *Va trimite cadrul pe toate porturile*
- *Va trimite cadrul pe toate porturile, cu excepţia portului 3*
- *Va ignora cadrul*
- *Va trimite cadrul pe portul 1*

Material cu drept de autor

3 Adresarea IP

Ce se învață din acest capitol?

- Specificațiile protocolului IP
- Noțiuni de *subnetting*
- Funcționarea protocolului ARP
- Funcționarea protocolului DHCP
- Configurări de rețea în Linux
- Configurări de rețea în Windows

Cine este...

Vint Cerf *este considerat "părintele Internetului". După ce a absolvit Universitatea Stanford, a lucrat la IBM, iar apoi s-a întors la Universitate la UCLA, unde a conectat primele două noduri a ARPAnet (predecesorul Internetului). A proiectat protocolul IP și a contribuit la dezvoltarea protocolului TCP. Din 2005, Vint Cerf lucrează la Google ca vicepreședinte.*

3.1 Prezentarea protocolului IP

Internetul a devenit o noțiune familiară pentru societatea din prezent. Cu toate acestea, în urmă cu 20 de ani prea puțini vizionari au intuit dezvoltarea pe care acesta urma să o cunoască. Multe dintre conceptele fundamentale ale infrastructurii IP de azi au fost definite în acea perioadă, precum formatul adresei IP, protocolul ARP, VLSM. Protocolul IP trebuia să răspundă schimbării paradigmei de comunicație de la o rețea cu câteva locații, precum rețeaua DARPA, la o rețea cu mii de locații cum era privit Internetul la mijlocul anilor `80. Apariția calculatoarelor personale și extinderea rețelei globale de comunicație dincolo centrele universitare au redefinit Internetul ca o rețea cu sute de milioane de noduri.

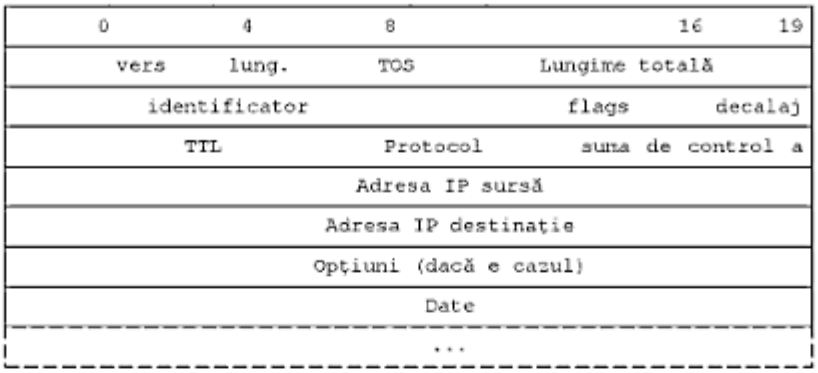
Versiunea 4 a protocolului IP a reușit să răspundă atât cerințelor de ierarhizare a spațiului de adrese impus de rețelele anilor `80, cât și cerințelor de scalabilitate ale Internetului actual. Pentru asigurarea scalabilității au fost standardizate protocoale menite să adreseze rutarea dinamică, translatarea de adrese, tunelarea pachetelor, etc.

Versiunea 6 a protocolului IP a fost inițial proiectată să asigure un spațiu de adrese mult mai generos, dar și un număr de servicii ce lipsesc din IPv4, precum QoS sau prelucrarea mai rapidă a pachetelor. Cu toate acestea, prelucrarea suplimentară presupusă de un antet de 40 de octeți față de unul de 20, precum și popularitatea deosebită de care se bucură IPv4 fac ca ponderea rețelelor IPv6 în structura actuală a Internetului să rămână de sub 5%. Prin urmare, pe parcursul acestei cărți, prin protocolul IP se va subînțelge doar referinrea la IPv4.

3.1.1 Structura antetului IPv4

Orice pachet ajuns la nivelul rețea este reîmpachetat, adăugându-i-se **antetul IP**. În **Error! Reference source not found.** sunt prezentate câmpurile ce compun antetul IP, urmând apoi o scurtă descriere a acestora.

Material cu drept de autor



3-1: Structura antetului IP

Din analiza antetului se identifică nu mai puţin de 10 câmpuri în afara celor ce precizează adresele destinaţie şi sursă. De-a lungul timpului semnificaţia acestor câmpuri a fost redefinită.

Câmpul versiune stabileşte versiunea IP folosită, antetul de IPv6 diferind de antetul IPv4.

Lungimea antetului este precizată explicit în cel de al doilea câmp în vederea flexibilizării dezvoltărilor ulterioare ale standardului IPv4, prin setări făcute în câmpul de opţiuni aflat în finalul antetului IP. Totuşi vasta majoritate a traficului în Internet foloseşte antete de lungime fixă, de 20 de octeţi, performanţele de referinţă ale echipamentelor de reţea (precum numărul de pachete comutate pe secundă) fiind calculate pentru trafic IP cu antet de lungime fixă.

Câmpul TOS (*Type of Service*) este folosit pentru implementarea unor politici distincte pentru tipuri de trafic diferit. Cea mai importantă utilizare a sa este pentru identificarea şi prioritizarea traficului de voce.

Câmpul de lungime totală este exprimat pe 16 biţi, rezultând o dimensiune maximă a cadrelor IP de 65535 de octeţi. După cum se poate observa din capitolul 7 [ref]??, nu există o dimensiune maximă pentru segmentele TCP, cea ce înseamnă că segmentele ce depăşesc 64 KB vor fi fragmentate la nivelul reţea. Deşi dimensiunea maximă prevăzută de standard este de 64KB, impunerea Ethernetului ca tehnologie dominantă pentru reţelele locale are drept consecinţă faptul că traficul TCP, după ce este segmentat în pachete de 64 KB la nivelul 4, va mai fi încă odată segmentat în cadre de 1500 octeţi la nivelul 3. Pentru a reduce complexitatea prelucrărilor asupra pachetelor, implementările curente ale stivei TCP/IP evită să realizeze două operaţii de fragmentare, impunând ca dimensiune maximă a cadrelor IP 1500 B şi nu 64 KB.

Mecanismul de secvenţiere a cadrelor reprezintă principalul mecanism de control al fluxului în TCP; cu toate acestea, se observă că un mecanism de secvenţiere există şi la nivelul antetului IP. Câmpul identificator stabileşte numărul datagramei şi este folosit în conjuncţie cu câmpul decalaj fragment pentru a reordona cadrele IP ajunse într-o altă ordine decât au fost transmise. Ambele câmpuri sunt în general stabilite de staţia ce emite pachetul, dar dacă pe calea către destinaţie mai are loc o fragmentare a pachetului valorile lor vor fi modificate.

Biţii de opţiune sunt folosiţi tot pentru a controla fragmentarea. Spre exemplu, bitul 50 din antetul IP este denumit bitul M sau bitul "*more fragments*". Acesta indică faptul că a avut loc o fragmentare şi că pachetul de faţă nu este ultimul din cadrul segmentului TCP. Bitul 51 este denumit Z sau bitul "*zero fragments*" şi are rolul de a semnaliza că pachetul actual este ultimul (sau singurul) din segmentul TCP.

Un câmp important din antetul IP este TTL (*Time To Live*), câmp ce defineşte numărul maxim de routere prin care un pachet poate să treacă. Principala sa funcţie este de a evita

Material cu drept de autor

ciclarea la infinit a unor pachete IP în cazul unor topologii cu bucle de rutare. O utilizare mai recentă a acestui câmp permite unui ISP să controleze conectarea unei stații, pentru o legătură dată. De exemplu, un ISP poate întrerupe conectivitatea atunci când pe o legătură în loc de o stație se conectează neautorizat un router ce are în spate o întreagă rețea locală.

Marea majoritate a traficului în Internet călătorește între sursă și destinație păstrând aceleași valori pentru câmpurile antetului IP, sigurul câmp modificat fiind câmpul TTL. Deși operația de decrementare a valorii câmpului TTL este una simplă, ea determină o încărcare semnificativă a routerului, deoarece în urma modificării acestui câmp va trebui să fie recalculată și suma de control a antetului. Suma de control se bazează pe un algoritm de redundanță ciclică (un algoritm CRC) ce are proprietatea că se poate verifica ușor, dar se calculează mult mai greu (verificarea se poate efectua fără a calcula explicit valoarea sumei de control).

Câmpul protocol specifică ce protocol a fost folosit pentru încapsularea de nivel transport. În figura de mai jos sunt prezentate câteva dintre valorile cele mai întâlnite ale acestui câmp. Valorile 4 și 41 sunt folosite în cazul tunelării iar valoarea 59 este folosită pentru a indica că nu mai există un alt antet, o astfel de conexiune fiind numită *IP raw*.

Valoare	Protocol
1	ICMP
4	IPv4
6	UDP
17	TCP
41	IPv6
59	Fără antet

3-2: Valorile câmpului protocol pentru antetul IP

3.1.2 Structura antetului IPv6

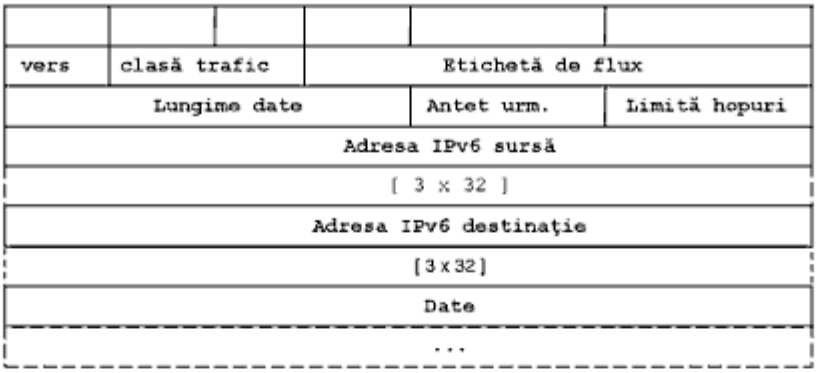
Protocolul IPv6 este standardizat prin RFC 2460, cele mai importante două diferențe față de IPv4 fiind lungimea fixă a antetului de 40 de octeți și eliminarea sumei de control a antetului.

Din structura unui pachet IPv6 se observă că cinci dintre câmpurile antetului IPv4 nu se mai regăsesc în antetul de IPv6: lungimea antetului, identificatorul de secvență, biți de control, decalaj fragment, suma de control a antetului.

Se observă că toate mecanismele de fragmentare din antetul IPv4 au fost eliminate. IPv6 realizează fragmentarea precum și alte funcții prin folosirea unor antete de extensie. Precizarea tipului de antet de extensie folosit se face prin câmpul „Antet următor" (*Next Header*), câmp ce folosește aceleași valori ca și câmpul „Protocol" din antetul IPv4 (vezi **Error! Reference source not found.**). Valoarea 43 a acestui câmp indică existența unui antet IPv6 de fragmentare după antetul curent.

Eliminarea sumei de control din antet este motivată de numărul mult mai mic al erorilor în rețelele actuale (în urma trecerii de la legăturile de cupru la cele optice sau prin folosirea cablurilor de cupru de o calitate mai bună). Rezultatul acestei modificari este creșterea vitezei de prelucrare a antetului de rețea, deoarece nu mai este necesar calculul sumei de control la fiecare modificare a câmpului „limită hopuri" (echivalentul câmpului TTL din Ipv4).

Material cu drept de autor



3-3: Structura antetului IPv6

Singurul câmp din antetul IPv6 ce nu are un echivalent în antetul IPv4 este câmpul „Etichetă de flux". Acest câmp permite routerelor să comute cadrele pe baza unei valori de 20 de biți și nu pe baza adresei destinație. Această metodă de comutare (folosind doar o etichetă de 20 de biți), înlesnește implementarea rețelelor IPv6/ MPLS (*MultiProtocol Label Switching*).

3.1.3 Clase de adrese

O adresă IP este un șir de 32 de biți ce identifică două lucruri: o rețea și o stație în cadrul acelei rețele.

Pentru a simplifica utilizarea adreselor IP se foloseşte formatul decimal. Astfel, o adresă IP dată: 10110001000001000001011000001000, se împarte mai întâi în grupuri de câte 8 biți: 10110001.00000100.00010110.00001000 și apoi fiecare grup este convertit în sistem zecimal: 177.4.22.8.

Deşi această nouă exprimare înlesneşte semnificativ lucrul cu adrese IP, aduce şi unele limitări în uşurința de a discerne porțiunea de rețea şi cea de stație din cadrul adresei IP, pentru cazurile în care sunt definite subrețele. Încercarea de a păstra reprezentarea zecimală ca model de referință pentru IP şi, în acelaşi timp de a pune în evidență distincția dintre cele două componente a dus la definirea claselor de adrese IP.

Odată cu definirea primelor trei clase pentru rutare a mai fost definit un spațiu de adrese folosit pentru adresarea multicast, anume clasa D. Restul adreselor vor constitui clasa E, reprezentând adrese rezervate. În 3-1 sunt prezentate cele cinci clase definite pentru spațiul de adrese IP.

Clasa	Primii biți	Nr. biți rețea	Nr. de rețele	Nr. biți stație	Nr. stații	Domeniul de valori
A	0...	8	2^7	24	2^{24}-2	1.0.0.0 – 126.255.255.255
B	10....	16	2^{14}	16	2^{16}-2	128.0.0.0 – 191.255.255.255
C	110...	24	2^{21}	8	2^8-2	192.0.0.0 – 223.255.255.255
D	1110...	Adrese multicast				
E	11110...	Rezervat				

3-1: Spațiul de adrese IP

Material cu drept de autor

Clasa A a fost proiectată pentru a satisface cerințele ridicate de rețelele de mari dimensiuni. Astfel, pentru definirea rețelei va fi folosit doar primul octet, pentru identificarea stației fiind disponibili 24 de biți, ceea ce oferă mai mult de 16,7 milioane de posibilități. În figura de mai sus se poate observa că domeniul de valori pentru clasa A nu include rețelele 0.0.0.0 și 127.0.0.0, acestea fiind rezervate. Clasa de adrese 0.0.0.0 nu este folosită datorită posibilelor confuzii cu rutele implicite, în vreme ce clasa 127.0.0.0 este rezervată pentru adrese de *loopback*, în scopul monitorizării și testării.

Tot din 3-1 se observă eliminarea a câte două adrese dintre cele ce pot fi alocate stațiilor, pentru fiecare dintre clasele rutabile. Cele două adrese sunt: adresa de rețea și adresa de difuzare.

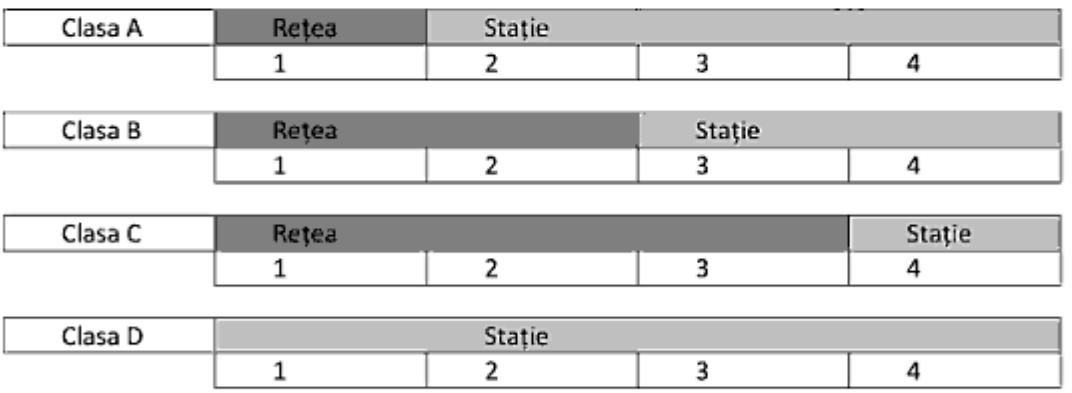
***O adresă IP de rețea** este o adresă pentru care toți biții de stație sunt 0.*

O astfel de adresă este folosită pentru identificarea întregii rețele. Aceasta este, de fapt, partea relevantă a oricărei adrese de stație ce călătorește peste Internet pentru toate routerele de pe parcurs.

***O adresă IP de difuzare** sau adresă de broadcast este o adresă pentru care toți biții de stație sunt 1. Un pachet destinat unei astfel de adrese va ajunge la toate stațiile din acea rețea.*

O clasă de adrese B este definită de valorile primilor doi biți din adresa IP, acești primi doi biți fiind 10. Din această constrângere rezultă că toate adresele IP ale căror prim octet se află între 10000000 și 10111111, adică între 128 și 191, aparțin unei clase B.

Câmpul de rețea pentru o clasă B va cuprinde primii doi octeți, dar deoarece primii doi biți ai primului octet sunt fixați, rămân doar 14 biți disponibili pentru a crea clase B. Pentru definirea stațiilor sunt folosiți ultimii doi octeți, adică 16 biți. Astfel pot fi obținute 16.384 rețele, fiecare având un număr maxim de 65.533 de stații.



3-2: Adresarea IP

Clasa C se definește prin alocarea primilor 3 octeți pentru definirea rețelei și doar a ultimilor 8 biți pentru identificarea stațiilor din aceeași rețea. Primii trei biți din primul octet trebuie să fie 110, adică valoarea acestui prim octet trebuie să se afle între 192 și 223 pentru ca o adresă să aparțină unei clase C.

Numărul rețelelor de clasă C depășește 2 milioane, fiecare dintre acestea putând să cuprindă 254 de stații.

Clasa de adrese D este folosită pentru rețele *multicast*. În decursul ultimilor 15 ani au existat numeroase standarde și propuneri de standardizare pentru asigurarea unei

Material cu drept de autor

infrastructuri de *multicast*, dar realitatea anului 2008 este că traficul de *multicast* reprezintă doar o foarte mică porțiune din traficul transferat în Internet. Cu toate acestea, convergența rețelelor de date cu cele de telefonie sau de televiziune oferă o notă de optimism în legătură cu viitorul comunicațiilor *multicast*. În Romania abia în anul 2006 a devenit disponibil comercial serviciul de trasmisiuni de *multicast*, un singur ISP oferind în acest moment acces la un M-Bone național.

Pentru adresa *multicast* spațiul de adrese este plat, toți cei 4 octeți fiind folosiți pentru definirea adresei de stație. Deoarece primii 4 biți ai primului octet sunt fixați, și anume 1110, numărul adreselor de *multicast* este de 268 milioane. Cu toate acestea au fost definite mai multe regiuni disjuncte, regiuni menite să servească obiective diferite: de la asigurarea integrării cu o infrastructură de *unicast*, până la definirea unor spații de adrese de *multicast* private. Primele 256 de adrese (cele cuprinse între 224.0.0.0 și 224.0.0.255) sunt definite ca aparținând zonei *Local Network Control Block*, aceastea fiind adresele folosite și de protocoalele de rutare: spre exemplu OSPF rezervă două dintre aceste adrese de *multicast*: 224.0.0.5 și 224.0.0.6 pentru procesul de alegere a routerului desemnat, iar RIPv2 folosește adresa 224.0.0.9 pentru trimiterea actualizărilor.

Clasa de adrese E este rezervată și nu poate fi folosită în rețelele publice sau în soluții de multicast.

3.1.4 Masca de rețea

Prin folosirea celor 3 clase rutate eficiența utilizării spațiului de adrese IPv4 este una extrem de redusă. Spre exemplu, pentru o rețea cu 4 noduri va fi alocată o clasă C, pierzându-se asftfel 250 de adrese. În cazul unei rețele de 300 de noduri alocarea unei clase B duce la pierderea a mai mult de 65.000 de adrese, și chiar prin reproiectarea rețelei și separarea sa în două rețele, se vor folosi două clase C, cea ce va duce la pierderea a peste 200 de adrese.

Protocolul IP impune ca orice adresă să conțină două informații: o adresă de rețea și adresa unei stații din cadrul acelei rețele. Separarea celor două câmpuri nu trebuie să apară la granița de octet. Pentru determinarea biților ce definesc adresa de rețea se folosește un șir de 32 de biți denumit mască de rețea.

***Masca de rețea** este un șir de 32 de biți care, în conjuncție logică cu o adresă IP, separă adresa de rețea, anulând biții de stație.*

Fiecare bit din masca de rețea ce corespunde (adică se află pe aceeași poziție) cu un bit din câmpul de rețea are valoare 1, în vreme ce toți biții corespunzători câmpului de stație au valoarea zero.

Exprimarea măștii de rețea poate fi realizată în forma zecimală sau sub forma unui prefix de rețea. În cazul exprimării zecimale cei 32 de biți sunt separați în grupuri de 8, apoi realizându-se conversia în zecimal. Procesul este unul similar cu exprimarea zecimală a adresei IP.

O altă reprezentare a măștilor de rețea este sub forma unui număr care indică numărul de biți de 1 consecutivi din masca de rețea. Acest tip de reprezentare poartă numele de **prefix de rețea**.

Pentru exemplificare, fie adresa IP: 141.85.37.133 și masca: 255.255.240.0. Masca de rețea este echivalentă cu prefixul /20. Pentru aceast exemplu câmpul de rețea va cuprinde primii 20 de biți, iar câmpul de stație ultimii 12. Adresa rețelei se obține prin operația de ȘI logic între mască și adresa IP: 141.85.32.0/20.

Adresa de difuzare se obține prin completarea tuturor biților din câmpul de stație cu valori de 1. Adresa de difuzare va fi: 141.85.63.255/20.

Material cu drept de autor

Pentru aceeaşi adresă, dar folosind prefixul /26 adresa de rețea ar fi: 141.85.37.128/26, iar adresa de difuzare: 141.85.37.191/26.

```
                          rețea                |       stație
1000 1101.0101 0101.0010 0101.10 |00 0101 - 141.85.37.133
1111 1111.1111 1111.1111 1111.11 |00 0000 - 255.255.255.192 - /26
-----------------------------------+------
1000 1101.0101 0101.0010 0101.10 |00 0000 - 141.85.37.128/26 - rețea
1000 1101.0101 0101.0010 0101.10 |11 1111 - 141.85.37.191/26 - difuzare
```

Este important de observat că aceaşi adresă poate fi adresă de stație sau adresă de difuzare în funcție de masca de rețea aleasă.

3.1.5 Subrețele

Totalitatea nodurilor ce pot comunica între ele folosind dispozitive de nivel fizic şi legătură de date (de exemplu: repetoare şi switchuri) definesc o rețea locală. Altfel spus, o rețea locală va cuprinde totalitatea echipamentelor de rețea ce pot comunica fără intermedierea unui router.

O rețea locală coincide cu un domeniu de difuzare. Astfel, toate stațiile din aceaşi rețea locală vor primi pachetele de *broadcast*.

Din motive de securitate, dar şi pentru optimizarea consumului de bandă în cadrul unei rețele locale, un administrator poate decide separarea unor secțiuni din rețea în subrețele diferite. Pentru asigurarea adresării va trebui să împartă spațiul inițial de adrese în mai multe secțiuni disjuncte.

Atenție! Distincția între „rețele" şi „subrețele" este una pur istorică. „Rețele" erau denumite doar spațiile de adrese ce corespundeau claselor A, B şi C. În prezent noțiunile sunt folosite interschimbabil.

Pentru a împărți spațiul de adrese 144.1.40.0/21 în două jumătăți se porneşte de la reprezentarea binară a spațiului inițial, apoi sunt delimitate câmpurile de rețea şi stație. Din câmpul de stație vor fi marcați un număr de biți pentru definirea de subrețele. Aceşti biți vor defini un nou câmp numit câmp de subrețea.

```
                rețea          |          stație
10010000.00000001.00101 |000.0000 0000 - 144.1.40.0/21
10010000.00000001.00101 |000.0000 0000 - 144.1.40.0/22 -prima subrețea
10010000.00000001.00101 |100.0000 0000 - 144.1.48.0/22 -a doua subrețea
                        ↖ subrețea
```

Pentru a împărți spațiul 144.1.48.0/22 în 5 subrețele, se caută cea mai apropiată putere a lui 2 egală sau mai mare cu numărul de subrețele căutat. Astfel pentru a obține 5 subrețele va trebui să împărțim spațiul de adrese în 8 secțiuni egale. Prefixul de rețea pentru fiecare dintre cele 8 subrețele va fi /25, adică prefixul spațiului inițial la care se adaugă numărul de biți necesar pentru a reprezenta cele 8 valori diferite.

```
                rețea          |          stație
10010000.00000001.001011 |00.0000 0000 - 144.1.40.0/22
10010000.00000001.001011 |00.0000 0000 - 144.1.40.0/25 -prima subrețea
10010000.00000001.001011 |00.1000 0000 - 144.1.48.0/25 -a doua subrețea
[...]
10010000.00000001.001011 |11.1000 0000 - 144.1.48.0/25 -a opta subrețea
                           ↖ subrețea
```

Material cu drept de autor

O dezbatere încă întâlnită în recomandările legate de alocarea adreslor IP este cea referitoare la folosirea primei şi ultimei subreţele. În lipsa precizării măştii de reţea, adresa primei subreţele poate fi confundată cu adresa spaţiului iniţial. În mod similar adresa de difuzare a ultimei subreţele poate fi confundată cu adresa de difuzare a spaţiului iniţial. Pentru exemplu de mai sus 144.1.48.0 poate fi ori adresa de reţea iniţială, dacă prefixul este /22, ori prima subreţea, dacă prefixul este /25. Adresa 144.1.51.255 este adresa de difuzare pentru întreg spaţiul iniţial pentru prefixul /22, sau adresa de difuzare a ultimei subreţele pentru /25.

Din păcate, evitarea folosirii primei şi a ultimei subreţele duce la o pierdere însemnată de adrese. Astfel, soluţia cea mai răspândită în reţelele actuale este de a folosi prima şi ultima subreţea, dar cu precizarea prefixului (sau a măştii de reţea) pentru orice adresă IP.

3.1.6 Super-reţele

Dimensiunea tabelei de rutare afectează atât latenţa procesului de găsire a căii optime, cât şi resursele hardware necesare pentru router (memorie, procesor). Pentru reducerea numărului de rute se poate folosi procesul de agregare a spaţiilor de adrese.

Agregarea de adrese este procesul invers împărţirii în subreţele.

În exemplul de mai jos sunt prezentate 4 spaţii de adrese alese special ca să difere doar prin cei mai puţin semnificativi doi biţi ai câmpului de reţea.

```
                  reţea                  |       staţie
1011 1110.0001 0100.0000 0100.0000 0000 - 190.20.4.0/24
1011 1110.0001 0100.0000 0101.0000 0000 - 190.20.5.0/24
1011 1110.0001 0100.0000 0110.0000 0000 - 190.20.6.0/24
1011 1110.0001 0100.0000 0111.0000 0000 - 190.20.7.0/24
-------------------------------------|-------------------------------
1011 1110.0001 0100.0000 0100.0000 0000 - 190.20.4.0/22
```

3-6: Agregarea a 4 clase C

Cele 4 clase din tabel sunt în fapt sferturile unui singur spaţiu de adrese. Adresa agregată, sau super-reţeaua ce cuprinde cele 4 clase, se obţine în acest caz reducând masca de reţea cu doi biţi. Aceşti doi biţi vor fi făcuţi zero, trecând în câmpul de staţie, pentru a determina adresa de reţea agregată.

Este important de precizat că deşi 190.20.4.0/22 este un spaţiu valid de adrese, nu poate fi folosit pentru alocarea de adrese într-o singură reţea. În alocarea adreselor nu se pot folosi super-reţele ale celor 3 clase rutate. Astfel, 140.20.4.0/22 este o subreţea din reţeaua de clasă B 140.20.0.0/16 şi poate fi folosit pentru alocarea într-o singură reţea, dar 190.20.4.0/22 este o super-reţea ce cuprinde 4 clase C, iar adrese din acest spaţiu pot fi alocate numai după o împărţire în subreţele.

Prefixul unei adrese IP valide nu poate fi mai mic decât prefixul clasei din care face parte respectiva adresă.

Nu orice două reţele pot fi agregate într-o super-reţea. Astfel, pentru a putea profita de această facilitate adusă de VLSM, alocarea adreselor trebuie făcută judicios nu doar în interiorul reţelei de către administratorul de reţea, ci şi la nivelul ISP-urilor şi chiar la nivel de ţară. Din păcate, în România avantajele reducerii tabelelor de rutare prin agregarea reţelelor, ca o consecinţă a alocării planificate a adreselor de reţea, au fost conştientizate extrem de târziu, astfel încât în tabelele de rutare ale marilor ISP-uri din România mai frecvent se întâlnesc prefixe de /26 decât prefixe /20, cum ar fi fost de aşteptat la o ţară de dimensiunile României.

Material cu drept de autor

3.1.7 ARP

În prezent protocolul de rezoluție a adresei – ARP este văzut adesea ca o componentă esențială a arhitecturii TCP/IP, dar lucrurile nu au stat dintotdeauna așa. Începutul anilor '80 a reprezentat o perioadă marcată de incertitudini în ceea ce privește standardizarea protocoalelor pentru rețelele de calculatoare. Dacă la nivelul rețelelor locale IEEE a reușit să reducă alegerea la trei standarde: Ethernet, Token Ring și Token Bus, comunicația între aceste rețele trebuia asigurată ori de IP ori de CLNS (**C**onnectionless **N**etwork **S**ervice). Nici anii ce au urmat nu au impus protocolul IP ca principalul câștigător la nivelul rețea de la începutul anilor '90, competiția desfășurându-se între IP și IPX sau Apple Talk.

Pentru legăturile punct-la-punct nu există nicio diferență între comunicația unicast și broadcast. Din acest motiv, pentru legăturile punct-la-punct nu este necesar un mecanism pentru determinarea adresei de nivel 2, folosindu-se doar adresa de difuzare. Ethernetul este însă un mediu multiacces, putând exista mai multe destinații în cadrul aceleași rețele locale.

ARP a fost standardizat de IETF în 1982 prin RFC 826 și reprezintă mecanismul pentru asigurarea comunicației unicast într-o infrastructură multiacces. Astfel, ARP și-a propus să ofere modalitatea de asociere a unei perechi <adresă de rețea, protocol de rețea> cu o adresă unică de nivel legătură de date. Deși standardul prevede posibilitatea funcționării ARP în conjuncție cu o varietate de protocoale de nivel rețea, în practică acesta a devenit o componentă integrantă a stivei de protocoale TCP/IP/Ethernet. Prin urmare, principala aplicabilitate a protocolului ARP a fost și rămâne determinarea corespondențelor între adresele IP și adresele MAC.

ARP se bazează pe construirea și menținerea unei **tabele ARP**. O tabelă ARP are rolul de a păstra corespondențele învățate între adresele IP și cele MAC. Acestea sunt construite dinamic și sunt stocate în memoria RAM. Deși există mecanisme pentru adăugarea statică sau eliminarea unei intrări într-o tabelă ARP, sunt rare situațiile în care un administrator de rețea va apela la ele.

Fiecare computer sau dispozitiv de rețea își păstrează propria sa tabelă ARP, în realitate existând câte o tabelă ARP pentru fiecare interfață activă. Astfel, un router cu trei interfețe Ethernet va menține trei tabele ARP distincte.

Cum funcționează ARP? Cum este construită tabela ARP?

Pentru a realiza configurațiile de rețea ale unei stații vor trebuit precizați minim patru parametri: adresa IP a stației, masca de rețea, adresa routerului implicit (default gateway) și adresa IP a serverului de DNS.

Serverul de DNS este folosit pentru a obține adresa IP a destinației pornind de la numele acesteia, spre exemplu o interogare de DNS va indica faptul că `www.cs.pub.ro` este asociat cu adresa 141.85.37.5.

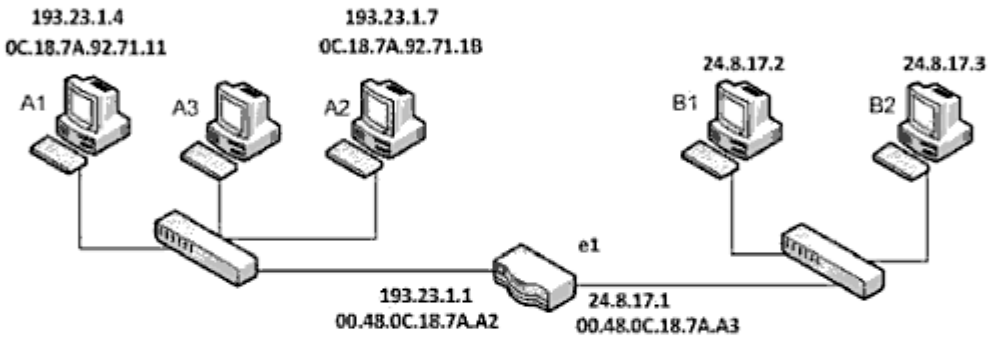
Datele de la nivelul aplicație vor fi prelucrate în conformitate cu operațiile specifice nivelurilor prezentare și sesiune (dacă e cazul) după care vor fi încapsulate la nivelul transport, precizându-se cel mai adesea tipul serviciului (portul sursă, portul destinație). Urmează încapsularea nivelului rețea care va atașa antetul IP, antet ce va conține informațiile legate de adresa IP sursă și adresa IP destinație, aceasta din urmă fiind în general obținută în urma unei rezoluții de DNS. Pentru construirea antetului de nivel legătură de date va trebui determinată adresa MAC destinație.

Adresele de nivel legătură de date au relevanță locală, nu și relevanță globală precum adresele de nivel rețea. Din acest motiv adresa MAC destinație din antetul de nivel doi va fi aceeași cu adresa MAC a destinației doar în cazul în care aceasta se află în aceeași rețea locală. Altminteri, din punctul de vedere al rețelei locale, adresa MAC destinație va fi adresa primului

Material cu drept de autor

router către destinație, deoarece orice router va mărgini rețeaua locală. Astfel, înainte de a căuta în tabela ARP, va trebui determinată care este următoarea destinație.

Pentru primul pas în procesul de rezoluție a adresei va trebui determinat dacă destinația se află în aceași rețea locală. Pentru aceasta se aplică masca de rețea atât adresei destinație cât și adresei sursă, iar dacă rezultatele operațiilor de ȘI logic coincid, se va considera că sursa și destinația se află în aceași rețea locală. În cazul acesta în tabela ARP va fi căutată direct adresa MAC a destinației, pornind de la adresa IP destinație. Dacă tabela ARP nu conține nicio intrare asociată cu adresa IP destinație, nodul sursă va temporiza (întârzia) încapsularea datelor și va crea un cadru nou, numit **cerere ARP**. Acest nou cadru va fi un cadru de difuzare la nivel legătură de date (deoarece adresa MAC a destinației nu este cunoscută), dar va avea în câmpul de date informații despre adresa IP destinație. Nodul destinație va identifica cadrul drept o cerere ARP, își va actualiza mai întâi tabela proprie, iar apoi va trimite un cadru, numit **răspuns ARP**, ce va fi *unicast* atât la nivel legătură de date, cât și la nivelul rețea. Pe baza acestui cadru sursa își va actualiza propria tabelă ARP va încapsula antetul de nivel legătură de date și va trimite cadrul.



3-7: Studiul ARP

Pentru mai multă claritate se va folosi folosi topologia din figura de mai sus pentru a urmări construirea tabelelor ARP.

Înainte de trecerea la nivelul legătură de date, adresa IP destinație va fi căutată în tabela ARP și nefiind găsită se va crea un cadru special (o cerere ARP) ce va avea în câmpul adresă destinație din antet adresa de difuzare: FF.FF.FF.FF.FF.FF, iar în câmpul adresă sursă adresa MAC a stației A1. În figura de mai jos este prezentată structura acestui cadru

Antet			Date				
MAC dest.	MAC sursă	Tip cadru	cod operație	MAC sursă	IP sursă	MAC dest.	IP dest
FFFF: FFFF: FFFF	0C18: 7A11: 7111	0x0806	1	0C18: 7A11: 7111	193.23. 1.4	0000: 0000: 0000	193.23. 1.7

3-8: Cerere ARP

Dacă se va considera că rețeaua din figură folosește *Ethernet* drept protocol de nivel legătură de date, datele vor fi difuzate și vor ajunge la A2, la A3 și la interfața routerului conectată la segmentul A. Antetul cadrului va fi analizat la nivelul legătură de date de către toți receptorii aflați în același domeniu de difuzare. Câmpul destinație fiind o adresă de difuzare, cadrul va fi trimis la nivelul superior. Cadrul este identificat drept o cerere ARP și

Material cu drept de autor

doar stația (interfața de rețea) a cărei adresă IP se regăsește în câmpul de date al cadrului va iniția un răspuns transmis ca unicast atât la nivel rețea, cât și la nivel legătură de date. Totodată, pe baza conținutului câmpului de date din cadrul de cerere ARP va fi creată prima intrare în tabela ARP a stației care s-a recunoscut ca și destinatar (în cazul de față, A2).

Antet			Date				
MAC dest.	MAC sursă	Tip cadru	cod operație	MAC sursă	IP sursă	MAC dest.	IP dest
0C18: 7A11: 7111	0C18: 7A92: 711B	0x0806	2	0C18: 7A92: 711B	193.23. 1.7	0C18: 7A11: 7111	193.23. 1.4

3-9: Răspuns ARP

După primirea răspunsului, A1 va putea insera în tabela sa ARP adresa MAC a lui A2, iar comunicația din acest moment va decurge fără probleme.

Fiind pe un segment *Ethernet*, toate cadrele schimbate de A1 și A2 vor ajunge la toate stațiile de pe segment, astfel că, deși nu au emis niciun cadru, atât A3 cât și routerul vor primi atât cererea ARP, cât și răspunsul. Cu toate acestea, nici cererea ARP, nici răspunsul nu vor duce la actualizarea tabelei ARP, cele două cadre fiind ignorate. Astfel tabelele celor două dispozitive rămân vide.

Protocolul ARP este un protocol de nivel legătură de date, iar pachetele sale sunt identificate folosind valoarea 0x0806 în câmpul Lungime/Tip. Această valoare este mai mare decât 0x0800, câmpul Lungime/Tip identificând tipul protocolului de nivel 2 și nu lungimea cadrului.

Câmpul cod operație din zona de date a cadrului ARP poate avea doar patru valori, două folosite de protocolul ARP și două de RARP. Astfel pentru valoarea 1 și 2 cadrul este interpretat ca o cerere, respectiv răspuns ARP, iar pentru valorile 3 și 4 este interpretat ca o cerere, respectiv răspuns RARP.

După popularea tabelei ARP va fi creat și antetul de nivel legătură de date al cadrului ce trebuia transmis inițial, după cum este prezentat și în 3-.

Antet 2			Antet 3		Date
MAC dest.	MAC sursă	Tip cadru	IP dest	IP sursă	
0C18: 7A92: 711B	0C18: 7A11: 7111	0x0800	193.23. 1.7	193.23. 1.4	

3-10: Cadrul de date

Cum are loc comunicația între stații aflate în rețele diferite?

S-a văzut că protocolul de rezoluție a adresei se bazează pe difuzări la nivel legătură de date. Routerele în schimb nu propagă pachetele de difuzare de nivel legătură de date în afara rețelei din care provin.

Revenind la primul pas al protocolului ARP, și anume la testul apartenenței la aceeași rețea a adresei IP sursă și a adresei IP destinație. Cu alte cuvinte, dacă rezultatul operației de ȘI logic între adresa sursă și masca de rețea diferă față de rezultatul operației de ȘI logic între adresa destinație și aceași mască de rețea, se concluzionează că sursa și destinația se află în rețele diferite. În acest caz, în antetul de nivel 2 va trebui precizată adresa următorului router

Material cu drept de autor

aflat pe calea către destinație, altfel spus, adresa routerului implicit (*default gateway*). Dacă în tabela ARP nu există o intrare pentru routerul implicit, atunci va fi trimis un cadru cerere ARP, pe adresa de difuzare de nivel 2, pentru a afla adresa IP a routerului implicit. Acesta va răspunde cererii, cu un cadru unicast ce va fi folosit pentru actualizarea tabelei ARP pe stația sursă. În cele din urmă va fi construit antetul de nivel 2 pentru cadrul de date, astfel încât adresa IP destinație va fi adresa IP a destinației finale, dar adresa MAC destinație va fi adresa MAC a routerului implicit.

În cazul rețelei de mai sus se consideră că stația A1 vrea să comunice cu B1. După operația IP(A1)&mască(A1) = IP(B1)& mască(A1), se determină că B1 nu se află în aceași rețea locală. Astfel A1 va căuta în tabela ARP o corespondență pentru adresa routerului implicit, adică pentru 193.23.1.1. Dacă această corespondență nu există, va trimite un cadru de cerere ARP dar care va avea precizat în câmpul de date ca adresă IP destinație 193.23.1.1. Cadrul fiind unul de difuzare, va fi recepționat de către toate dispozitivele de rețea aflate pe acest segment. A2 și A3 vor ignora cadrul, deoarece acesta va avea precizată ca adresă IP destinație altă valoare decât adresele lor. Routerul va trimite un cadrul de răspuns ARP similar cu cel din 3-[ref], în care MAC sursă va fi: 00.48.0C.18.7A.A2, iar IP sursă va fi 193.23.1.1.

Pe baza cadrului de cerere ARP, routerul își va actualiza propria tabelă ARP corespunzătoare interfeței dinspre segmentul A, iar apoi pe baza cadrului de răspuns A1 își va adăuga în tabela ARP o intrare nouă, ce face corespondența între 193.23.1.1 și adresa MAC a interfeței routerului: 00.48.0C.18.7A.A2. Din acest moment stația A1 va încapsula transmisia destinată stației B1 folosind adresa IP a lui B1 (24.8.17.2) și adresa MAC a interfeței e0 a routerului (00.48.0C.18.7A.A2).

Adresa destinație va folosi routerului pentru a determina interfața pe care trebuie trimis pachetul și astfel procesul de rutare va determina că pachetul trebuie trimis pe interfața e1 a routerului. Routerul va face mai întâi testul dacă interfața pe care trebuie trimis pachetul este în aceași rețea cu destinația finală a pachetului. În cazul de față IP(e1)&mască(e1) va da același rezultat cu IP(B1)&mască(e1), astfel va fi căutată în tabela ARP a interfeței e1, o corespondență pentru adresa IP a lui B1. Dacă această corespondență nu există va fi trimisă o cerere ARP ce va conține adresa IP destinație a stației B1 (a destinației finale).

Dacă în schimb B1 nu ar fi fost în aceași rețea cu interfața e1, ar fi fost extrasă din ruta folosită adresa următorului hop, și căutată în tabela ARP o corespondență pentru adresa următorului hop.

Pentru topologia din figură[ref], în urma a două procese de cerere/răspuns ARP și o rescriere a antetului de nivel 2 operată de router, pachetul va ajunge la destinație, această comunicație simplă fiind realizată prin trimiterea a nu mai puțin de 6 cadre cu antete de nivel 2 diferite. În plus, în tabela ARP a stației A1, a interfeței e0, a interfeței e1 și a stației B1 a fost adăugată câte o înregistrare

Cum are loc comunicația între stații aflate în rețele diferite dacă nu s-a precizat adresa routerului implicit?

Pentru sistemele de operare ce rulează la nivelul stațiilor, lipsa adresei routerului implicit echivalează cu limitarea comunicației la rețeaua locală. Pe de altă parte, în cazul routerelor ce au ca interfață de ieșire o rețea de tip multiacces (de exemplu *Ethernet*), dar nu au precizată și adresa următorului router trebuie căutat un alt mecanism pentru a asigura ieșirea din rețeaua locală. Un caz similar este și cel al unor dispozitive dedicate ce rulează sisteme de operare monolitice, cu implementări parțiale ale stivei TCP/IP datorită resurselor hardware mult mai limitate decât în cazul calculatoarelor personale (de exemplu mașini de marcat, automate de cafea, etc.). Atât pentru rute incomplet specificate, cât și pentru implementări parțiale ale stivei TCP/IP nu va mai exista diferență între comunicația între noduri din aceași rețea locală și

Material cu drept de autor

comunicația între noduri aflate în rețele diferite. Stațiile nu vor mai avea nevoie decât de precizarea adresei IP, pentru orice adresă IP destinație urmând să inițieze o cerere ARP.

Soluția se bazează pe rularea la nivelul routerului de ieșire din rețeaua locală a serviciului de *proxy ARP*.

Proxy ARP este o extensie a protocolului de rezoluție a adresei. Pornind de la faptul că routerul nu va transfera pachetele de difuzare, *Proxy ARP* va determina routerul să răspundă la toate cererile ARP destinate unor adrese în afara rețelei cu adresa MAC a interfeței conectate în acea rețea.

Este important de subliniat că, deși pentru o adresă IP dată nu poate exista mai mult de o singură intrare în tabela ARP, mai multe adrese IP pot fi asociate cu o singură adresă MAC, acest fapt făcând posibilă funcționarea comunicației prin *Proxy ARP*.

În topologia folosită anterior, pentru a permite comunicația între A1 și B1 folosind proxy ARP, testul de apartenență în aceași rețea nu mai poate fi făcut la nivelul stației, deoarece aceasta nu mai are disponibilă o mască de rețea. A1 va iniția un cadru de cerere ARP, ce va avea ca adresă IP destinație B1 (și nu adresa IP a interfeței `e0`). Cererea va ajunge la toate stațiile conectate în rețeaua locală, dar A2 și A3 o vor ignora nerecunoscând adresa IP destinație. Routerul în schimb, rulând *proxy ARP*, va testa mai întâi dacă cererea ARP este destinată unei stații aflate în afara rețelei din care provine. Testul va folosi masca și adresa interfeței pe care a fost primită cererea, precum și adresa IP destinație. Cum IP(B1)&mască(e0) este diferit de IP(e0)&mască(e0), routerul va decide că destinația se află în altă rețea. În acest caz routerul va trimite cadrul de răspuns ARP folosind ca adresă sursă de nivel rețea adresa destinației finale (în cazul de față adresa lui B1 - 24.8.17.2), și adresa de MAC a interfeței de ieșire din rețea, adică 00.48.0C.18.7A.A2. Totodată, routerul își va adăuga în tabela ARP a interfeței e0 corespondența între 0C.18.7A.11.71.11 și 193.23.1.4, iar A1 își va adăuga în tabela ARP intrarea ce asociază 00.48.0C.18.7A.A2 cu 24.8.17.2.

Cadrul de date va fi încapsulat apoi folosind tabela ARP, precizând ca adresă IP destinație 24.8.17.2, iar ca adresă MAC destinație 00.48.0C.18.7A.A2, exact ca și în cazul folosirii unui router implicit.

Router implicit vs. Proxy ARP?

Spre deosebire de *Proxy ARP*, în care cererea ARP este adresată stației destinație, în cazul precizării routerului implicit cererea ARP este adresată direct routerului. În cazul proxy ARP stațiile se comportă ca și cum toate destinațiile s-ar afla în rețeaua lor locală, având ca adresă MAC adresa routerului. Aceasta înseamnă că dacă o stație vrea să transmită către trei stații aflate în rețele diferite, stația sursă va emite trei cereri ARP (câte una pentru fiecare). Cererile vor fi interceptate și li se va răspunde de către router; aceasta duce la o creștere a traficului, precum și a dimensiunii tabelei ARP de la nivelul stației. În cazul *default gateway* stația sursă va testa apartenența destinațiilor la rețeaua proprie și în cazul în care observă că ele fac parte din altă rețea, stația sursă nu va trimite cereri ARP direct către ele ci vor folosi adresa MAC a routerului implicit (pe care o pot afla trimițând o singură cerere ARP). *Proxy ARP* încarcă routerul, care trebuie să răspundă la cererile ARP destinate stațiilor din afara rețelei; precizarea routerului implicit încarcă stațiile, care trebuie să testeze apartenența stațiilor destinație la rețeaua locală.

Deși pare firească întrebarea care dintre cele două metode este mai bună, în rețelele locale competiția s-a încheiat în favoarea metodei bazate pe folosirea routerului implicit. Stațiile de lucru au devenit foarte puternice în decursul ultimilor 15 ani, astfel încât distribuirea la nivelul stațiilor a testului de apartenență a sursei și destinației la același LAN aduce acestora o încărcare nesemnificativă, eliberând routerul de procesul decizional asociat cu *Proxy ARP*.

Material cu drept de autor

Pe de altă parte, încărcarea routerului la rularea *Proxy ARP* nu este semnificativă, mai ales pentru un router ce conectează o rețea locală. Din acest motiv majoritatea routerelor (toate routerele CISCO spre exemplu) vor avea activat implicit *Proxy ARP*.

În cazul unei rețele locale cu mai mult de o singură ieșire de Internet, precizarea routerului implicit oferă un control mult mai strict al stațiilor, și permite implementarea balansării pe bază de sursă a traficului.

Dacă s-ar analiza strict doar cele două protocoale, concluzia ar fi că în cazul în care stațiile comunică preponderent cu alte stații din cadrul aceleași rețele locale comunicația bazată pe folosirea routerului implicit va fi lentă, datorită testului suplimentar, în vreme ce pentru cazul unei rețele în care majoritatea traficului părăsește rețeaua locală Proxy ARP va emite câte o cerere ARP pentru fiecare adresă destinație diferită.

3.1.8 DHCP

DHCP (*Dynamic Host Configuration Protocol*) este un protocol client-server prin intermediul căruia serverul furnizează stației client parametrii de configurare necesari funcționării într-o rețea. DHCP oferă, de asemenea, posibilitatea controlului accesului la rețeaua locală (pe criteriul adresei fizice), precum și mobilitate, mutarea dintr-o rețea în alta fiind posibilă fără reconfigurarea manuală a gazdei.

DHCP furnizează un mecanism prin care serverul atribuie adrese IP clienților. Există trei modalități de alocare a adreselor IP: alocare dinamică, manuală sau automată.

Alocarea dinamică presupune definirea unui set de adrese IP. Adresele IP alocate sunt înlăturate din mulțimea adreselor disponibile, dar în momentul expirării perioadei de închiriere (dacă nu este prelungit contractul de închiriere) acestea se pot întoarce în zona adreselor disponibile pentru ca apoi să fie alocate unui alt nod de rețea. Perioada de închiriere a adreselor IP variază în funcție de implementarea serverului de DHCP, valori uzuale fiind 24 sau 192 de ore.

Alocarea manuală presupune definirea pe server de asocieri între adrese MAC și adrese IP. La primirea unei cereri DHCP, adresa MAC sursă va fi căutată în lista de asocieri. Dacă nu există o asociere definită, în funcție de configurație, serverul poate ignora cererea, sau poate trece în modul de alocare dinamic. Alocarea manuală permite administratorului implementarea unor politici de control al accesului la rețea, fiind una dintre primele recomandări de securizare a rețelei locale. În același timp, permite un grad de flexibilitate ridicat în cazul schimbărilor de topologie precum apariția unui nou server de DHCP sau schimbarea routerului de ieșire din rețeaua locală (*default gateway*).

Alocarea automată îmbină simplitatea de configurare a alocării dinamice (trebuie doar definit setul de adrese IP disponibile, și nu o listă de asocieri MAC-IP) cu avantajele de securitate ale alocării statice: în cazul unui rețele cu număr de adrese disponibile egal cu cel al stațiilor o nouă stație nu va putea primi parametrii de rețea, deoarece o adresă IP alocată nu se mai întoarce în mulțimea adreselor disponibile decât la restartarea serviciului de DHCP.

Funcționând în rețeaua locală, DHCP nu necesită un serviciu orientat pe conexiune care să ofere controlul traficului, detectarea și rectificarea erorilor sau secvențierea corectă a datelor. Un astfel de serviciu (TCP) ar introduce încărcare și întârzieri nejustificate. De aceea, se folosesc datagrame UDP, pe portul 67 pentru server și pe portul 68 pentru client.

Conversația dintre client și server constă în următorii pași:

1. La pornire, stația client DHCP trimite cereri pentru inițierea comunicației cu serverele DHCP. Aceste cereri sunt trimise prin intermediul mesajelor BROADCAST de tip DHCPDISCOVER.
2. La primirea cererilor, serverul determină dacă o poate onora. În caz afirmativ, serverul răspunde cu mesaj UNICAST de tip DHCPOFFER, care poate include adresa IP, masca de rețea,

Material cu drept de autor

adresa gateway, adresa serverului de nume, precum şi perioada de valabilitate. În caz negativ, cererea poate fi transmisă mai departe, către un alt server DHCP (DHCP *relay*).

3. Dacă oferta este acceptată de către client, acesta va trimite un mesaj BROADCAST de tip DHCPREQUEST, în care sunt ceruți parametrii respectivi. Se trimite un mesaj de tip BROADCAST şi nu unul de tip UNICAST pentru a stabili care server a fost ales, în cazul în care DHCPDISCOVER a ajuns la mai multe servere. În implementările uzuale ale DHCP, stația începe să folosească adresa IP alocată, deşi procesul de confirmare încă nu s-a încheiat.
4. Serverul ales trimite un mesaj de confirmare UNICAST, de tip DHCPACK. În cazul în care adresa a fost alocată până la primirea DHCPREQUEST, serverul va trimite un mesaj DHCPNACK, procesul reluându-se de la pasul 1.

3.2 Definirea parametrilor de rețea în Linux

3.2.1 Configurarea temporară

Linux pune la dispoziție două utilitare pentru configurarea interfețelor de rețea, configurare ce se va pierde după repornirea sistemului. Primul dintre acestea este `ifconfig`. Prezent pe toate platformele Unix, acest utilitar permite stabilirea adresei IP, a măştii de rețea, precum şi a adresei de difuzare.

În exemplul de mai jos este prezentată folosirea comenzii fără precizarea explicită a măştii de rețea. În acest caz va fi configurată masca de rețea a clasei din care face parte adresa, pentru exemplul de față 255.255.255.0, precum şi adresa de difuzare 192.1.3.255.

```
# ifconfig eth0 192.1.3.2
```

Dacă se doreşte atribuirea adresei 192.1.3.1/26 vor trebui precizate explicit atât masca de rețea, cât şi adresa de difuzare:

```
# ifconfig eth0 192.1.3.1 netmask 255.255.255.252 broadcast 192.1.3.63
```

Comanda `ifconfig` mai poate fi folosită atât pentru inspectarea configurațiilor curente de rețea, a parametrilor de funcționare: MTU, număr de pachete trimise, număr de pachete primite, etc

```
# ifconfig eth0
eth0 Link encap:Ethernet HWaddr 00:0A:95:E2:04:D4
inet addr:10.38.252.237 Bcast:10.38.255.255 Mask:255.255.0.0
inet6 addr: fe80::20a:95ff:fee2:4d4/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:553119 errors:0 dropped:0 overruns:0 frame:0
TX packets:904848 errors:0 dropped:0 overruns:0 carrier:0
 [...]
```

Pentru fiecare interfață fizică de rețea pot fi definite mai multe interfețe logice, denumite **subinterfețe**. Din punct de vedere logic, fiecare subinterfață a unui router reprezintă o interfață distinctă. Din acest motiv două subinterfețe nu pot avea adrese IP din aceeaşi subrețea. În exemplul de mai jos sunt definite adresa IP, masca de rețea şi adresa de difuzare pentru interfața eth0:1, adică pentru una dintre subinterfețele interfeței eth0:

```
# ifconfig eth0:1 42.1.3.1 netmask 255.255.0.0 broadcast 42.1.255.255
```

Toți parametrii de rețea se resetează la oprirea interfeței ce poate fi făcută specific pentru o interfață sau prin repornirea serviciului de rețea, cea ce duce la reinițializarea tuturor interfețelor de rețea. Configurațiile temporare de rețea se vor pierde în cazul dezactivării interfeței. În cazul unei subinterfețe dezactivarea este echivalentă cu ştergerea sa.

```
# ifconfig eth0:1 down
# /etc/init.d/rc.d/networking restart
```

Începând de la versiunea de kernel 2.2 a apărut un pachet de utilitare pentru manipularea configurărilor de rețea, cunoscut sub numele **iproute**. Ajuns la cea de a doua versiune

Material cu drept de autor

(iproute2), el reprezintă o alternativă puternică, permiţând realizarea de setări foarte sofisticate.

Pentru configurarea parametrilor de reţea se foloseşte utilitarul `ip addr`, parte a pachetului `iproute`. În cazul apelării comenzii doar cu adresa IP, va fi configurată masca 255.255.255.255, altfel spus prefixul de reţea /32:

```
# ip addr add dev eth0 192.168.38.11
# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 1000
link/ether 00:0a:95:e2:04:d4 brd ff:ff:ff:ff:ff:ff
inet 192.168.38.11/32 scope global eth0
```

Astfel, pentru configurarea unui interfeţe este necesară precizarea adresei IP, a prefixului de reţea, precum şi a adresei de difuzare:

```
# ip addr add dev eth0 192.168.38.11/24 broadcast 192.168.38.255
# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 1000
link/ether 00:0a:95:e2:04:d4 brd ff:ff:ff:ff:ff:ff
inet 192.168.38.11/24 brd 192.168.38.255 scope global eth0
```

Pentru a defini configuraţii pentru subinterfeţe este folosită opţiunea `label`. Din motive de compatibilitate, se recomandă ca numele etichetelor să înceapă cu numele interfeţei. Astfel, pentru `eth0` etichete valide sunt: `eth00`, `eth0.0`, `eth0:0`, `eth0-test`. Pentru că `ifconfig` vede etichetele ca pe o interfaţă virtuală, el nu poate rula decât cu cele de tip `interfaţă:număr`.

```
# ip addr add dev eth0 192.168.38.11/24 broadcast 192.168.38.255 label eth0:0
# ip addr show label eth0:0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 1000
link/ether 00:0a:95:e2:04:d4 brd ff:ff:ff:ff:ff:ff
inet 192.168.38.11/24 brd 192.168.38.255 scope global eth0:0
```

Pentru activarea sau dezactivarea unei interfeţe, sau ştergerea configuraţiilor logice asociate cu o subinterfaţă se foloseşte un alt utilitar al pachetului `iproute2: ip link`. De exemplu, pentru a închide interfaţa `eth0` se poate folosi comanda:

```
ip link set eth0 down
```

Utilitarul `ip link` mai poate fi util pentru schimbarea parametrilor de nivel legătură de date (adresă MAC, MTU, etc), precum şi la afişarea acestor parametri:

```
# ip link show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 1000
link/ether 00:0a:95:e2:04:d4 brd ff:ff:ff:ff:ff:ff
```

Pe lângă funcţiile de manipulare a tabelei de rutare şi de definire a tunelelor, `iproute2` oferă suport şi pentru de politici de trafic (*traffic shaping*).

3.2.2 Configurarea permanentă

Pentru configurarea permanentă a parametrilor de reţea este folosit fişierul `/etc/network/interfaces`.

Acest fişier este specific distribuţiei Debian (în alte distribuţii locaţia sa poate fi diferită) şi conţine informaţiile necesare pentru configurarea interfeţelor de reţea. Programele care efectiv utilizează acest fişier sunt `ifup` şi `ifdown` şi sunt rulate din `/etc/init.d/networking`, script responsabil de configurarea reţelei, în procesul de pornire al sistemului de operare.

Pentru definirea statică a parametrilor de reţea administratorul poate specifica: adresa IP, masca de reţea, adresa *default gateway*, adresa serverului de nume, etc, sau doar o parte dintre aceşti parametri. De asemenea, pot fi specificate alte acţiuni care să fie realizate atunci când interfaţa este pornită, respectiv oprită:

```
auto eth0
```

Material cu drept de autor

```
iface eth0 inet static
address 192.168.1.3
netmask 255.255.255.0
gateway 192.168.1.1
dns-nameservers 192.168.1.2
up echo $IFACE up
down echo $IFACE down
up route add -host 192.168.38.100 gw 192.168.1.17
```

În exemplul de mai sus, interfeţei `eth0` îi este atribuită adresa 192.168.1.3/24, adresa routerului implicit este 192.168.1.1, serverul de DNS 192.168.1.2. La ridicarea (activarea) interfeţei va fi afişat mesajul „`eth0 up`" şi va fi adăugată în tabela de rutare o rută către reţeaua 192.168.38.100/32 (reţea formată dintr-o singură adresă) cu următorul hop 192.168.1.17.

Pentru obţinerea dinamică a parametrilor de reţea este editat tot fişierul `/etc/network/interfaces`. Spre exemplu, pentru a configura dinamic interfaţa `eth0`:

```
auto eth0
iface eth0 inet dhcp
```

Orice modificări permanente vor fi aplicate doar în urma repornirii serviciului de reţea:

```
#/etc/init.d/networking restart
```

3.3 Configurarea serviciului DHCP pe un server Linux

3.3.1 Instalarea şi configurarea serverului DHCP

Serverul DHCP primeşte cereri de la clienţii din reţea şi furnizează acestora parametrii IP necesari funcţionării corespunzătoare.

Pentru instalarea versiunii 3 a serverului se foloseşte pachetul `dhcp3-server`. Pachetul `dhcp` conţine versiunea 2 a serverului de DHCP:

```
#apt-get install dhcp3-server
```

Odată configurat, serverul poate fi manipulat prin intermediul scriptului de iniţializare:

```
#/etc/init.d/dhcp {start | stop | reload | force-reload}
```

sau folosind comanda `dhcpd`, ca utilizator privilegiat.

Comanda `dhcpd` permite specificarea altor parametri de funcţionare ai serverului, spre exemplu, numărul portului pe care să primească cereri (conform RFC 2131, acesta este 67):

```
#dhcpd -p NUMAR_PORT
```

Specificarea altui port decât cel implicit poate fi utilă în procesul de depanare sau în securizarea reţelei.

Fişierul de configurare al serverului DHCP este `/etc/dhcpd.conf`. De asemenea, fişierul `/var/lib/dhcp/dhcpd.leases` păstrează baza de date a clienţilor DHCP pentru serverul respectiv.

Fişierul de configurare păstrează informaţii despre clienţii din reţea. În cadrul lui, pot fi declarate opţiuni globale, aplicabile tuturor clienţilor, sau opţiuni pentru fiecare client sau grup de clienţi.

Considerând acest aspect, există două tipuri de specificaţii în cadrul fişierului `dhcpd.conf`: parametrii şi declaraţii.

Declaraţiile sunt folosite pentru a descrie topologia reţelei sau clienţii, şi precizează adresele care pot fi atribuite clienţilor. Există patru tipuri de declaraţii care conturează topologia reţelei: `host`, `goup`, `subnet` şi `shared-network`.

Material cu drept de autor

Declarația `host` este folosită atunci când se dorește specificarea parametrilor de funcționare pentru o anumită gazdă (de exemplu, alocarea statică a adresei IP, în funcție de adresa MAC).

Declarația `group` folosită atunci când se dorește gruparea mai multor clienți specificați cu declarația `host`. Acești clienți vor avea un set comun de parametri.

Declarația subnet trebuie făcută pentru fiecare subrețea deservită de serverul de DHCP.

Declarația `shared-network` este folosită în cazul în care există mai multe subrețele declarate cu `subnet`, care vor avea un set comun de parametri de configurare.

Atunci când serverul DHCP primește o cerere, va consulta mai întâi declarația `host` a clientului (dacă există), după care declarația `group` (dacă există), apoi declarația `subnet` pentru subrețeaua din care a venit cererea. Urmează declarația `shared-network`, în final fiind consultați parametrii globali.

Parametrii precizează dacă trebuie realizată o acțiune (de exemplu, dacă serverul DHCP trebuie să furnizeze adresa unui client neidentificat), cum trebuie realizată o acțiune (de exemplu, cât timp un client poate păstra o anumită adresă IP) sau parametrii de configurare care trebuie furnizați clientului (de exemplu, specificarea serverului DNS care va fi folosit).

În cadrul parametrilor, pot exista opțiuni. Opțiunile sunt parametri facultativi specificați folosind cuvântul cheie `option`. Restul parametrilor fie specifică modul de funcționare al serverului, fie sunt obligatorii, conform protocolului DHCP .

Mai jos sunt prezentați câțiva parametri des întâlniți în configurarea DHCP:

```
group{
    option routers 192.168.1.1;
    option subnet-mask 255.255.255.0;
    option broadcast-address 192.168.1.255;
    option domain-name-server 141.87.37.2;
}
```

- `option domain-name-servers ADRESA [, ADRESA1 ...]`: specifică serverul (serverele) de nume pe care le va folosi clientul;
- `option subnet-mask MASCA_DE_RETEA`: masca de rețea care va fi furnizată clienților;
- `option broadcast-address ADRESA`: adresa de broadcast furnizată clienților;
- `option routers ADRESA`: adresa default a *gateway*-ului;

```
host gazda1{
    hardware ethernet 00:02:55:F3:12:F0;
    fixed-address 192.168.1.3;
}
```

- `hardware HARDWARE-TYPE HARDWARE-ADDRESS`: parametru folosit pentru recunoașterea clienților;
- `HARDWARE-TYPE`: implementate *Ethernet* și *Token-Ring* (nu mai e folosit).
- `fixed-address ADRESA [, ADRESA1 ...]`: parametru folosit pentru atribuirea statică a uneia sau mai multor adrese IP.

```
subnet 192.168.1.0 netmask 255.255.255.0{
    [parametri]
    range 192.168.1.3 192.168.1.34;
}
```

- `range ADRESA1 ADRESA2`: intervalul din care vor fi atribuite dinamic adrese IP clienților, pentru o anumită subrețea:

```
allow unknown-clients
deny unknown-clients
```

- `allow/deny`: controlează comportamentul serverului DHCP în cazul în care primește cereri de la clienți necunoscuți.

Un fișier de configurare poate avea structura următoare:

```
shared-network reteaua1{
    [parametri de retea]
```

Material cu drept de autor

```
        group{
            [parametri de grup]

            subnet ADRESA1 netmask MASCA1{
                [parametri de subretea]
            }
            subnet ADRESA2 netmask MASCA2{
                [parametri de subretea]
            }

            host NUME{
                [parametri de host]
            }
        }

        host NUME1{
            [parametri de host]
        }
    }
```

Fișierul poate conține spații, linii goale sau tab-uri adiționale pentru formatare. Sintaxa este *case-insensitive*, comentariile începând cu # și sfârșindu-se cu linie nouă.

Schimbările făcute fișierului de configurare vor avea efect după restartarea serverului:

```
/etc/init.d/dhcp restart
```

Un alt fișier pentru configurarea serverului DHCP este `/etc/default/dhcp`. Aici pot fi specificate, de exemplu, interfețele pe care serverul să asculte cereri DHCP (folosind declarația `INTERFACES`).

3.3.2 DHCP Relay

DHCP Relay permite transmiterea cererilor DHCP dintr-o subrețea în care nu există server DHCP către unul sau mai multe servere DHCP din alte subrețele.

Pentru instalare, este necesar pachetul *dhcp-relay:*

```
apt-get install dhcp-relay
```

Agentul DHCP poate fi configurat la pornire, prin parametrii transmiși comenzii `dhcrelay`, sau prin intermediul fișierului `/etc/default/dhcp-relay`.

În acest fișier se pot specifica:

- interfețele pe care agentul să primească cereri DHCP: declarația INTERFACES;
- serverele DHCP către care să fie trimise cererile DHCP primite: declarația DHCP_SERVERS.

3.3.3 Exemplu de configurare DHCP

Este prezentat în continuare un exemplu de configurare DHCP.

Serverul (stația MPLS4) are două plăci de rețea, fiecare dintre acestea făcând parte dintr-o subrețea, și va furniza adrese IP gazdelor din subrețeaua 192.168.1.0/24 din intervalul 192.168.1.10 - 192.168.1.20, iar gazdelor din subrețeaua 10.16.200.0/24 din intervalul 10.16.200.5-10.16.200.150.

Două din stațiile din rețea (MPLS2 și MPLS3) vor obține adrese IP și parametri de configurare, prima în mod dinamic, iar cea de-a doua în mod static.

Interfețele serverului DHCP sunt configurate astfel:

```
root@MPLS4:/etc# cat network/interfaces
auto lo
iface lo inet loopback
# Prima interfata
auto eth0
iface eth0 inet static
name Ethernet LAN card
address 192.168.1.1
broadcast 192.168.1.255
netmask 255.255.255.0
network 192.168.1.0
# A doua interfata
auto eth1
```

Material cu drept de autor

```
iface eth1 inet static
name Ethernet LAN card
address 10.16.200.1
broadcast 10.16.200.255
netmask 255.255.255.0
network 10.16.200.0
```

Fişierul de configurare al serverului este:

```
root@MPLS4:/etc# cat dhcpd.conf
option domain-name "test.ro";
option domain-name-servers 192.168.1.1;
option subnet-mask 255.255.255.0;
default-lease-time 600;
max-lease-time 7200;
# Grupul celor doua subretele
# Au in comun timpii de valabilitate
group{
    # Timpul default de valabilitate pentru acest grup
    default-lease-time 2000;

    # Timpul maxim de valabilitate pentru acest grup
    max-lease-time 8000;

    # Prima subretea
    subnet 192.168.1.0 netmask 255.255.255.0{
        option routers 192.168.1.1;
        option broadcast-address 192.168.1.255;
        range 192.168.1.10 192.168.1.20;
    }

    # A doua subretea
    subnet 10.16.200.0 netmask 255.255.255.0{
        option routers 10.16.200.1;
        option broadcast-address 10.16.200.255;
        range 10.16.200.5 10.16.200.150;
    }
    # Gazda configurata static
    host gazda1{
        hardware ethernet 00:02:55:F3:12:F0;
        fixed-address 192.168.1.3;
    }
}
```

Cele două staţii date ca exemplu sunt configurate după cum urmează:

```
root@MPLS-2:/etc/network# cat interfaces
auto lo
iface lo inet loopback
# The primary network interface
auto eth0
iface eth0 inet dhcp
name Ethernet LAN card

root@MPLS3:/etc# cat /etc/network/interfaces
auto lo
iface lo inet loopback
# The primary network interface
auto eth0
iface eth0 inet dhcp
name Ethernet LAN card
```

La pornirea staţiilor, sau când este repornit serviciul de reţea, acestea vor iniţia o conversaţie cu serverul DHCP în urma căreia vor obţine adresele IP şi parametrii de configurare.

```
root@MPLS3:/etc# /etc/init.d/networking restart
Setting up IP spoofing protection: rp_filter.
Reconfiguring network interfaces...ifup: interface lo already configured
Internet Software Consortium DHCP Client 2.0pl5
[...]
Listening on LPF/eth0/00:02:55:f3:12:f0
Sending on LPF/eth0/00:02:55:f3:12:f0
Sending on Socket/fallback/fallback-net
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPNAK from 192.168.1.1
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 8
DHCPOFFER from 192.168.1.1
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPACK from 192.168.1.1
bound to 192.168.1.3 -- renewal in 1000 seconds.
done.
```

Material cu drept de autor

În cazul stației MPLS3 se observă că a fost atribuită adresa 192.168.1.3, deoarece această stație are adresa fizică 00:02:55:f3:12:f0, specificată cu parametrul `host` în fișierul de configurare al serverului.

```
root@MPLS-2:/etc/network# /etc/init.d/networking restart
Setting up IP spoofing protection: rp_filter.
Reconfiguring network interfaces...
ifup: interface lo already configured
[...]
Listening on LPF/eth0/00:02:55:f3:0c:02
Sending on LPF/eth0/00:02:55:f3:0c:02
Sending on Socket/fallback/fallback-net
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPNAK from 192.168.1.1
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 5
DHCPDISCOVER on eth0 to 255.255.255.255 port 67 interval 7
DHCPOFFER from 192.168.1.1
DHCPREQUEST on eth0 to 255.255.255.255 port 67
DHCPACK from 192.168.1.1
bound to 192.168.1.11 -- renewal in 1000 seconds.
done.
```

În cazul stației MPLS2 adresa atribuită este următoarea adresă liberă din intervalul de adrese specificat, respectiv 192.168.1.11 .

3.4 *Configurarea adreselor de rețea în* Windows Server 2008

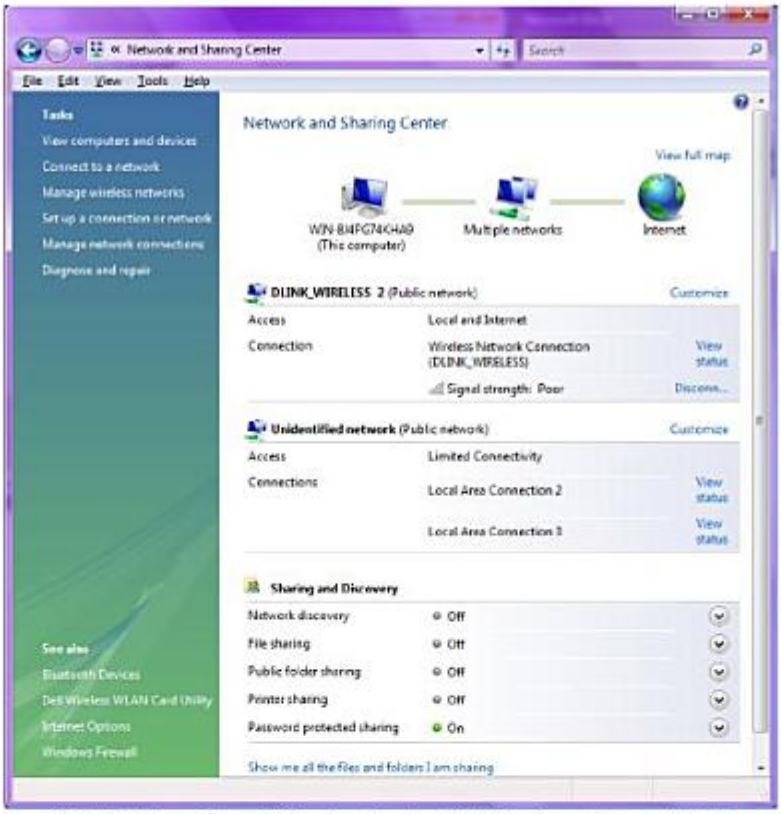
Windows Server 2008 prezintă două zone principale în care este posibilă configurarea parametrilor de rețea: *Network and Sharing Center* și *Network Connections*.

3.4.1 Network and Sharing Center

Network and Sharing Center reprezintă principalul utilitar de configurare a rețelei în *Windows Server 2008*. El poate fi accesat în unul dintre următoarele moduri:

- Din meniul *Start*, clic dreapta pe *Network* și se selectează *Properties*;
- În *System Tray* (denumit și *Notification Area*), dacă este afișată pictograma de conectivitate la rețea, printr-un clic pe aceasta urmat de selecția opțiunii *Network and Sharing Center* din meniu;
- Din *Control Panel*, accesând *Network and Internet* urmat de *Network and Sharing Center* (sau direct accesând *Network and Sharing Center* pentru *Control Panel* în modul *Classic View*).

Material cu drept de autor



3-11: Network and Sharing Center în *Windows Server 2008*

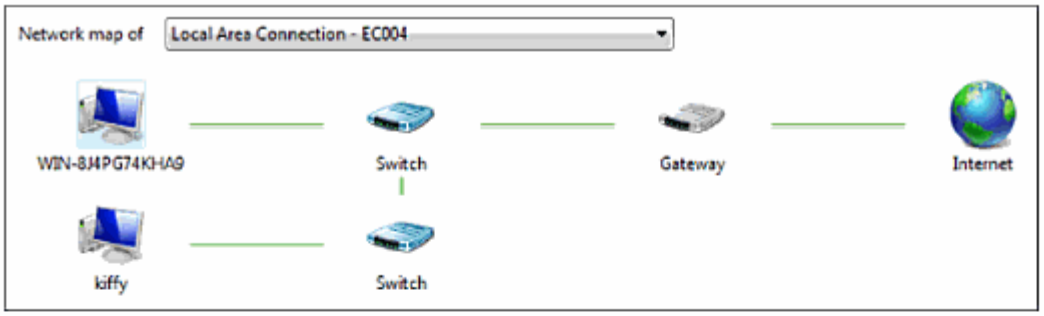
Atât în cazul rețelelor cablate cât şi pentru cele *wireless*, *Network and Sharing Center* atribuie una dintre cele trei locații posibile: *public*, *private* şi *domain*. Aceste locații reprezintă parametri ce sunt setați pentru orice calculator ce rulează Windows *Vista* sau *Windows Server 2008*, fiecare configurându-şi apartenența la rețea printr-una dintre cele trei locații. Diferite proprietăți ale rețelei pot fi activate sau dezactivate automat în funcție de tipul ei.

Implicit, toți clienții sunt membri ai unei locații de tip *public*. Pentru un astfel de calculator, *Windows Firewall* este activ, serviciul de *Network Discovery* este oprit, partajarea fişierelor şi a imprimantei este dezactivată iar generarea unei hărți a rețelei (*Network Map*) este indisponibilă.

Când un calculator este asignat unei locații de tip *private* serviciul de *Nework Discovery* şi *Network Map* sunt activate. Partajarea fişierelor este oprită, dar, spre deosebire de locația *public*, partajarea poate fi activată individual şi independent pe fiecare calculator.

Dacă un calculator devine membru al unui domeniu *Active Directory*, el este automat inclus în locația de tip *domain*. Membrii acestei locații beneficiază de aproximativ aceeaşi configurație ca şi cei din tipul de locatie *private*, cu excepția că parametrii *Windows Firewall*, *Network Discovery* şi *Network Map* sunt determinate de politicile de grup ale domeniului (*Group Policy Settings*).

Material cu drept de autor



3-12: Exemplu de *Network Map*

O schiță a rețelei (*Network Map*) permite afișarea în mod grafic a echipamentelor conectate la rețeaua locală și a modului în care acestea sunt interconectate, precum și legătura la Internet.

Pentru ca un *Network Map* să poată fi generat sunt necesare două componente în rețea:

- *Link Layer Topology Discovery* (*LLTD*) *Mapper* este componenta care interoghează celelalte dispozitive din rețea și care generează *Network Map*-ul.
- *LLTD Responder* este componenta interogată, care răspunde cererilor venite de la *LLTD Mapper*

Cele două componente sunt instalate implicit doar pe Windows *Vista* și *Windows Server 2008*, dar este posibilă instalarea unui *LLTD Responder* și pe un calculator ce rulează Windows XP, permițându-i acestuia să fie „văzut" de către un *LLTD Mapper* din rețea ce generează un *Network Map*.

Alte opțiuni disponibile în *Network and Sharing Center*:

- ***Network Discovery***: Permite calculatorului propriu să poată localiza alte calculatoare din rețea și să poată fi localizat, la rândul său. Opțiunea poate fi setată pe *On*, *Off* sau poate avea valoarea *Custom*, spre exemplul în situația în care *Network Discovery* este activ dar *firewall*-ul nu deține o regulă pentru a permite funcționarea sa în rețea.
- ***File Sharing***: Partajarea fișierelor creează automat o permisiune în *firewall* pentru ca protocolul să poată funcționa. Activarea *File Sharing*-ului permite utilizatorilor să partajeze fișierele din propriul profil, adică din `%systemroot%\Users\%username%`. Administratorii de sistem pot partaja orice fișier din calculator.
- ***Public Folder Sharing***: În directorul de profil al fiecărui utilizator există un subdirector numit *Public* care este automat partajat în momentul activării acestei opțiuni. La activarea *Public Folder Sharing* este activată automat și opțiunea de *File Sharing*.
- ***Printer Sharing***: Opțiunea oferă posibilitatea de partajare a accesului la imprimantele instalate local, pentru a putea fi folosite de orice alt calculator din rețea. De asemena, activarea aceste opțiuni are ca efect și activarea opțiunii de *File Sharing*.
- ***Password Protected Sharing***: Opțiunea este disponibilă doar pe sistemele care nu sunt membre ale unui domeniu. În momentul activării sale, accesul la resursele locale partajate este restricționat doar pentru cei care au un cont valid de utilizator pe calculatorul gazdă.

3.4.2 Network Connections

Windows Server 2008 detectează și configurează automat conexiunile asociate interfețelor de rețea din sistem. Aceste conexiuni sunt listate în *Network Connections*, alături de alte conexiuni configurate manual, cum ar fi cele de tip *dial-up*, VPN-uri sau conexiuni de tip PPPoE.

Network Connections poate fi accesat în mai multe moduri:

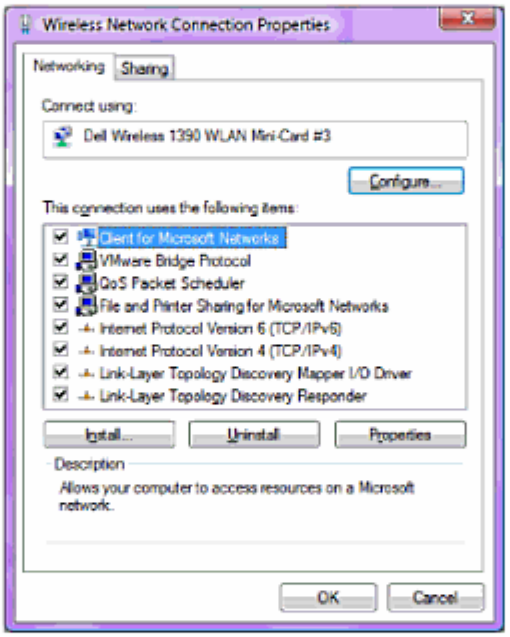
- Din *Server Manager*, clic pe *View Network Connections*;

Material cu drept de autor

- Din fereastra *Initial Configuration Tasks* (afişată de la prima lansare a sistemului, după instalare), clic pe *Configure Networking*;
- Din interfaţa *Network and Sharing Center*, clic pe *Manage Network Connections*;
- De la meniul *Start*, scriind comanda `ncpa.cpl` sau `control netconnections` fie în câmpul de *Search*, fie la *Run*.

Conexiunile în sine nu permit calculatoarelor să comunice printr-o reţea. În realitate, clienţii, serviciile şi protocoalele în contextul conexiunilor sunt cele care permit comunicaţia între două sau mai multe staţii. În fereastra de proprietăţi a conexiunilor sunt afişaţi clienţii, protocoalele şi serviciile ataşate acelei conexiuni. Una dintre modalităţile de a afişa proprietăţile unei conexiuni este din *Network Connections*, prin clic dreapta pe una dintre conexiuni şi apoi clic pe *Properties*, din meniu. De asemenea, se poate ajunge aici şi din *Network and Sharing Center*, prin clic pe *View Status* şi apoi pe *Properties*, în dreptul conexiunii dorite.

Elementele bifate indică componente ce sunt ataşate conexiunii respective:



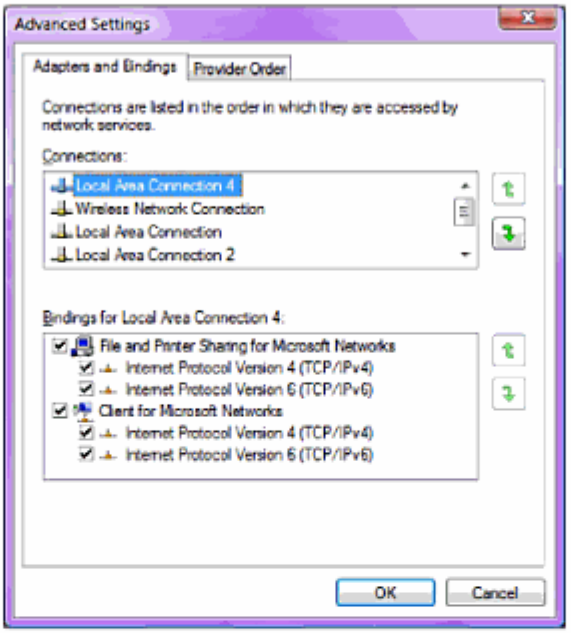
3-13: Fereastra de proprietăţi ale unei conexiuni

- ***Network Clients***: Într-o reţea Windows, clienţii sunt componente software care permit unei staţii să se conecteze cu un anumit sistem de operare din reţea. Implicit, singurul client disponibil pentru toate conexiunile locale este *Client for Microsoft Networks*. Acesta permite calculatoarelor ce rulează Windows să se conecteze şi să partajeze resurse între ele.
- ***Network Services***: Serviciile sunt componente software ce oferă funcţionalităţi suplimentare conexiunilor. *File and Printer Sharing for Microsoft Networks* şi *QoS Packet Scheduler* sunt două dintre serviciile ataşate implicit tuturor conexiunilor locale. *File and Printer Sharing for Microsoft Networks* permite calculatorului să partajeze fişiere pentru a fi accesate din reţea. *QoS Packet Scheduler* oferă control asupra traficului din reţea, cu posibilitatea de a prioritiza anumite fluxuri de date şi servicii.
- ***Network Protocols***: Calculatoarele comunică printr-o conexiune doar prin intermediul protocoalelor ataşate acelei conexiuni. Suportul pentru IPv4 şi IPv6 este inclus implicit pentru

Material cu drept de autor

toate conexiunile locale, alături de suportul pentru *LLTD Mapper* şi *LLTD Responder* (descrise în secţiunea 3.4.1).

Este posibilă vizualizarea setărilor avansate pentru conexiunile din *Network Connections*. Pentru aceasta, din fereastra *Network Connections*, se alege opţiunea *Advanced Settings* din meniul *Advanced*[1]:



3-14: Setări avansate ale conexiunilor din Network Connections

În fereastra de setări avansate, la categoria *Adapters and Bindings* sunt afişate conexiunile curente, în ordinea priorităţilor. Modificarea ordinii conexiunilor, deci ajustarea priorităţilor, are ca efect forţarea sistemului de operare în încercarea de a comunica prin anumite conexiuni în ordinea definită de administrator. De asemenea, se poate configura şi ordinea ataşării serviciilor pentru fiecare conexiune în parte, în partea de jos a ferestrei.

3.4.3 Vizualizarea parametrilor de reţea

Configuraţia IP a unei interfeţe presupune cel puţin o adresă IPv4 şi o mască de reţea sau o adresa IPv6 şi un prefix de reţea. Pe lângă aceste configuraţii minimale, se pot regăsi şi informaţii ca *default gateway*, adresa unuia sau a mai multor servere DNS, un sufix DNS şi adrese ale serverelor WINS.

Una dintre cele mai simple comenzi ce pot fi folosite pentru a consulta configuraţia IP a interfeţelor din sistem este `ipconfig`, ce poate fi introdusă fie în *prompt*-ul de comandă fie în *PowerShell*. Pentru a obţine informaţii extinse despre toate interfeţele instalate în sistem se poate folosi parametrul `/all`:

```
PS C:\Users\Administrator> ipconfig /all
[...]
Ethernet adapter Local Area Connection 4:
```

[1] În Windows *Vista* este posibil ca meniul (File, Edit, View, etc.) să nu fie afişat implicit. În acest caz, se ţine apăsată tasta Alt şi acesta va apărea.

Material cu drept de autor

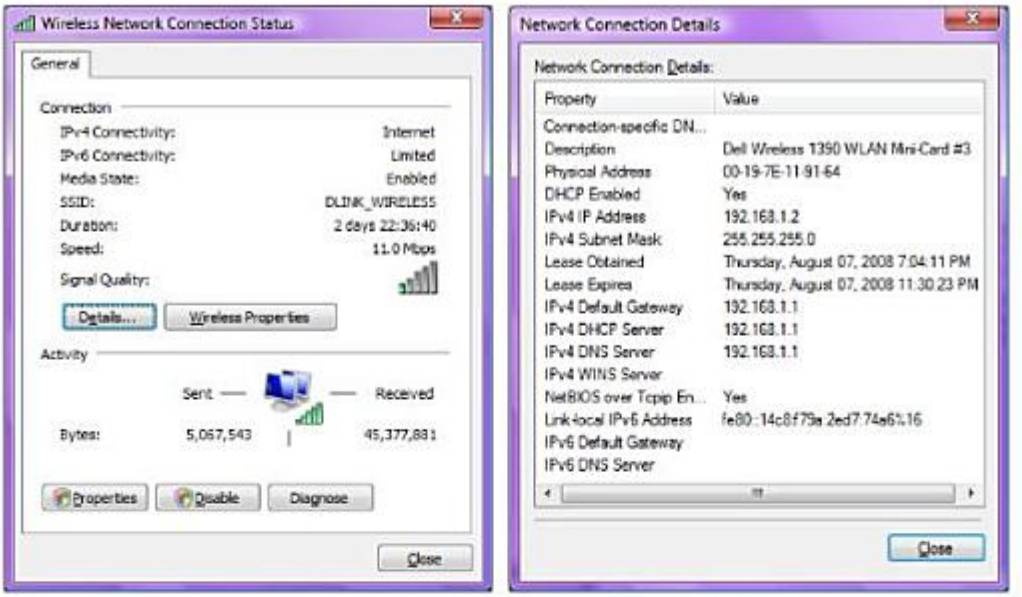
```
   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Bluetooth LAN Access Server Driver
   Physical Address. . . . . . . . . : 00-19-7D-E1-AC-04
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Wireless Network Connection:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Dell Wireless 1390 WLAN Mini-Card
   Physical Address. . . . . . . . . : 00-19-7E-11-91-64
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::14c8:f79a:2ed7:74a6%16(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.1.2(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Thursday, August 07, 2008 7:04:11 PM
   Lease Expires . . . . . . . . . . : Thursday, August 07, 2008 9:48:52 PM
   Default Gateway . . . . . . . . . : 192.168.1.1
   DHCP Server . . . . . . . . . . . : 192.168.1.1
   DHCPv6 IAID . . . . . . . . . . . : 268441982
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-0F-CE-4B-E1-00-19-B9-5E-FA-DE
   DNS Servers . . . . . . . . . . . : 192.168.1.1
   NetBIOS over Tcpip. . . . . . . . : Enabled

Ethernet adapter Local Area Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : fiberlink.rdsnet.ro
   Description . . . . . . . . . . . : Broadcom 440x 10/100 Integrated Controller
   Physical Address. . . . . . . . . : 00-13-D4-9E-50-C8
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
[...]
```

3-15: Fragment din rezultatul comenzii *ipconfig /all*

Comanda `ipconfig` poate fi folosită şi pentru a forța primirea unei configurații prin DHCP, dacă există un astfel de server în rețea. Pentru aceasta, comanda `ipconfig /release` şterge configurația dinamică de pe toate interfețele configurate dinamic iar comanda `ipconfig /renew` trimite cereri DHCP pe toate interfețele ce au fost setate pentru configurare automată.

Vizualizarea stării unei conexiuni poate fi realizată şi prin intermediul utilitarelor *Network Connections* şi *Network and Sharing Center*. Pentru a afişa fereastra de stare a unei conexiuni se apasă clic dreapta pe una dintre conexiunile din *Network Connections* şi se alege *Status* din meniul contextual (sau dublu clic direct) sau se apasă direct pe *View Status* din dreptul conexiunii dorite, din *Network and Sharing Center*.

Material cu drept de autor



3-16: Starea unei conexiuni, împreună cu detaliile sale

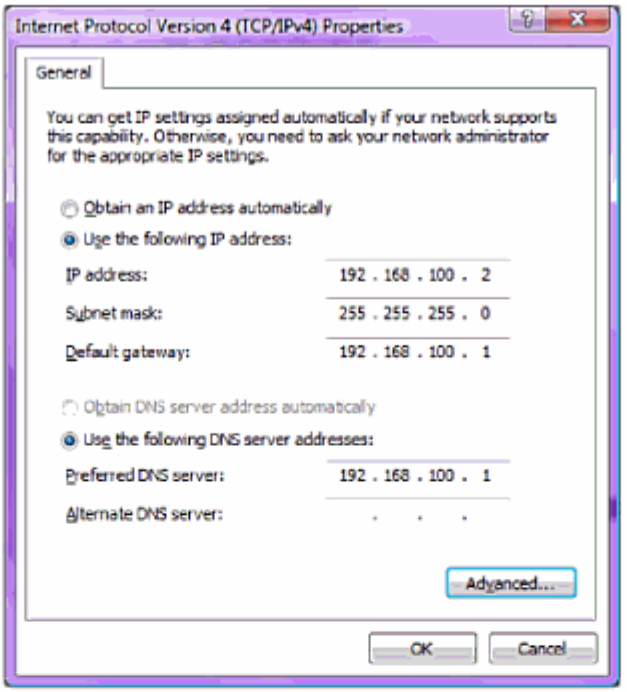
Din fereastra de stare a conexiunii se poate accesa şi o fereastră cu detalii despre conexiune, ce include detalii despre interfața de rețea folosită, adresele configurate şi alte detalii legate de configurarea automată (dacă este cazul).

3.4.4 Configurarea manuală a adreselor IP

O conexiune poate beneficia de o configurație IP manuală sau automată. Configurația manuală este denumită şi configurație statică deoarece persistă şi după restartarea sistemului şi este de importanță critică pentru servere şi echipamente specializate într-o rețea.

Asignarea manuală a unei adrese statice şi a altor parametri de configurare IPv4 unei conexiuni se face folosind fereastra *Internet Protocol Version 4 (TCP/IP) Properties* din lista de protocoale ataşate unei conexiuni. Pentru a o accesa, se deschide fereastra de proprietăți a unei conexiuni (figura 3-13) şi se face dublu clic pe *Internet Protocol Version 4 (TCP/IPv4)*.

Material cu drept de autor

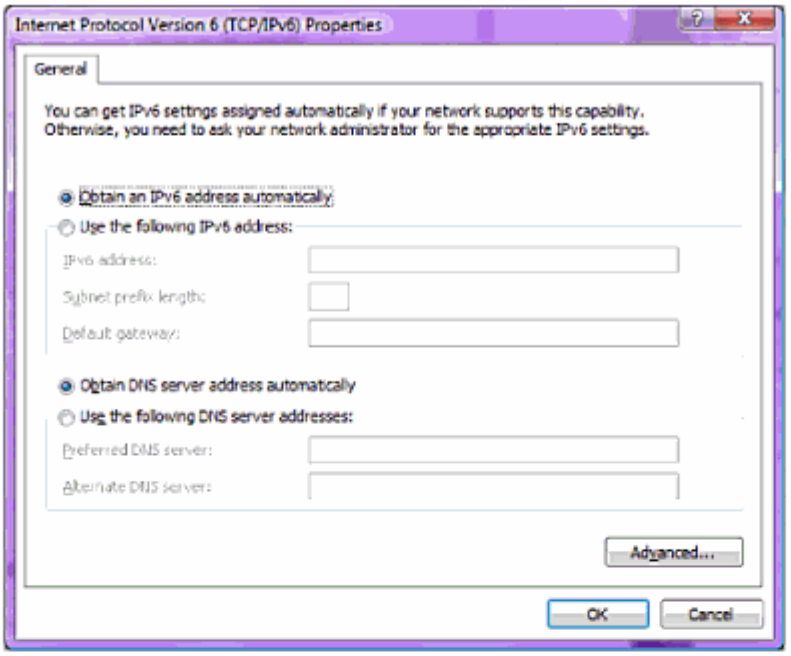


3-17: Fereastra de configurare a adreselor protocolului IPv4

Implicit, o conexiune de rețea este setată pentru a-şi obține automat configurația. Pentru a specifica o configurație statică, este necesară selectarea opțiunii *Use the following IP address* eventual împreună cu specificarea unui server DNS primar şi a unuia alternativ.

În cazul utilizării IPv6, de cele mai multe ori nu este necesară configurarea de adrese IPv6 static pe stații ci doar pe routere, stațiile obținându-şi informațiile prin autoconfigurare. Totuşi, *Windows Server 2008* permite asignarea de adrese IPv6 şi la nivel de stații. Pentru configurarea unei adrese IPv6, procedura este aceeaşi ca şi la IPv4, cu diferența că din fereastra de proprietăți a unei conexiuni se alege *Internet Protocol Version 6 (TCP/IPv6)*.

Material cu drept de autor



3-18: Fereastra de configurare a adreselor protocolului IPv6

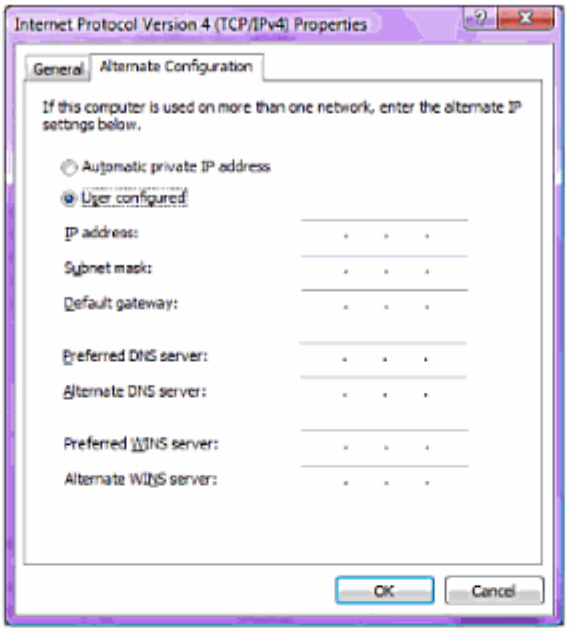
Atenţie, introducerea unei adrese statice de tip IPv6 necesită şi introducerea unei adrese a unui server DNS de tip IPv6.

3.4.5 Definirea unei configuraţii IP alternative

În cazul în care în aria de *broadcast* a unui client nu este localizat un server DHCP, un client ce a fost configurat să îşi obţină configuraţia IP în mod automat va recurge la informaţiile din configuraţia IP alternativă, dacă aceasta a fost definită.

Asignarea unei configuraţii alternative se face prin selectarea paginii *Alternate Configuration* din ferestrea de configurare *Internet Protocol Version 4 (TCP/IPv4) Properties*. Configuraţia alternativă suportă specificarea unei adrese IP, a unei măşti de reţea, a unui *default gateway*, a unuia sau a două servere DNS şi a unuia sau a două servere WINS.

Material cu drept de autor



3-19: Fereastra pentru configurația IPv4 alternativă

Deoarece o configurație alternativă permite unui calculator să folosească o configurație IP specifică și detaliată în momentul în care nu se detectează un server DHCP în rețeaua locală, ea este utilă pentru sistemele mobile care circulă între rețele, unele cu servere DHCP iar altele fără.

Pentru IPv6 nu se poate specifica o configurație alternativă.

3.4.6 Asignarea automată a adreselor private (APIPA)

APIPA este un acronim pentru *Automatic Private IP Addressing* și reprezintă o facilitate de asignare automată a adreselor locale în rețele temporare sau *ad hoc*. Când un calculator ce rulează Windows a fost configurat să își obțină configurația IP în mod automat, dacă nu există un server DHCP în rețeaua locală și nici configurația alternativă nu a fost specificată, el va folosi APIPA pentru a-și asigna o adresă privată din intervalul 169.254.0.1 – 169.254.255.254 (se observă că masca de rețea este 255.255.0.0).

În mod implicit, toate calculatoarele sunt setate să folosească APIPA în cazul în care nu primesc răspuns de la un server DHCP din rețeaua locală. Mai exact, după cum se observă din figura 3-19, în fereastra de configurare alternativă este bifată, implicit, opțiunea *Automatic Private IP Address* ceea ce evidențiază atât lipsa unei configurații alternative cât și utilizarea APIPA. Practic, se poate considera că, în lipsa unui server DHCP, este folosită întotdeauna configurația alternativă, chiar și în cazul în care aceasta specifică folosirea APIPA.

APIPA este o tehnică utilă pentru că ea permite calculatoarelor aflate în același domeniu de *broadcast* să comunice chiar și în lipsa unui server DHCP sau a oricărui alt tip de configurație manuală. De asemenea, ea reprezintă o soluție alternativă și pentru situația în care un server DHCP devine nefuncțional[1]. Dacă la un moment de tip ulterior serverul DHCP redevine operațional, configurația automată va dobândi prioritate înaintea celei setate de

[1] Totuși, dacă un server DHCP devine neoperațional, stațiile vor recurge la APIPA doar după expirarea timpului de închiriere a ultimei configurații obținute de la acesta, deci trecerea nu se va face instantaneu.

Material cu drept de autor

APIPA şi o va rescrie, staţiile continuându-şi comunicaţia folosind schema de adresare oferită de serverul DHCP[1].

În practică, se poate detecta momentul în care APIPA a intrat în funcţiune dacă două sau mai multe calculatoare din reţea pot comunica între ele dar nu şi cu altele sau în afara reţelei. O practică recomandată în acest caz este verificarea configuraţiei IP a staţiilor pentru a indentifica prezenţa adreselor oferite de APIPA şi verificarea funcţionării şi a accesului la serverul DHCP.

Există şi o serie de limitări importante ce trebuie avute în vedere în momentul în care staţiile sunt configurate prin APIPA. Spre exemplu, calculatoarele configurate astfel vor putea comunica doar cu alte calculatoare configurate prin APIPA din acelaşi domeniu de *broadcast* (din considentele adresării IP în interiorul unei reţele locale, vor fi acceptate doar pachetele care aparţin reţelei calculate din adresa IP şi masca configurată pe interfaţă). De asemenea, staţiile ce folosesc APIPA nu vor putea avea acces la Internet şi nu se pot configura adrese pentru servere DNS sau WINS şi nu se poate specifica niciun *default gateway*.

```
Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::525:fc37:994d:636f%14
   IPv4 Address. . . . . . . . . . . : 169.254.216.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
```

3-20: Exemplu de configuraţie APIPA (fragment rezultat *ipconfig*)

APIPA reprezintă varianta Microsoft de *Zero Configuration*. În mod general, tehnica pentru IPv4 poartă numele de adresare tip IPV4LL (*IPv4 Link Local*). Mai multe detalii despre adresele locale automate IPv4 pot fi consultate în RFC 3927[2] iar pentru IPv6 în RFC 4862[3].

3.5 Configurarea din linia de comandă

Pentru inspectarea şi modificarea configuraţiei IP din linie de comandă, *Windows Server 2008* pune la dispoziţie utilitarul *netsh*. Comanda poate fi folosită ca un *prompt* de comandă, introducând doar `netsh` în `cmd.exe` sau în *PowerShell* (ca mai jos) sau se poate scrie fiecare comandă cu parametrii completi pentru a primi imediat un rezultat.

```
PS C:\Users\Administrator> netsh
netsh>
```

În general, navigarea prin comenzile disponibile în `netsh` se poate face treptat, în sensul că la fiecare adăugare a unui parametru, dacă acesta nu constituie o comandă completă, `netsh` va afişa o listă cu toţi parametrii suportaţi în continuare, împreună cu o scurtă explicaţie a lor. Spre exemplu, dacă se doreşte vizualizarea tuturor comenzilor de tip `show`, se poate introduce următoarea comandă:

```
PS C:\Users\Administrator> netsh interface ipv4 show

The following commands are available:

Commands in this context:
show addresses - Shows IP address configurations.
show compartments - Shows compartment parameters.
show config    - Displays IP address and additional information.
show destinationcache - Shows destination cache entries.
show dnsservers - Displays the DNS server addresses.
show dynamicportrange - Shows dynamic port range configuration parameters.
```

[1] Explicaţia pentru acest comportament stă în faptul că o staţie care recurge la APIPA o va face pentru că este setată să obţină o configuraţie automată şi nu a reuşit acest lucru. Deci, practic, înlocuirea configuraţiei APIPA cu cea prin DHCP se face deoarece configuraţia primară are prioritate în faţa celei alternative.

[2] http://tools.ietf.org/html/rfc3927

[3] http://tools.ietf.org/html/rfc4862

Material cu drept de autor

```
        show global    - Shows global configuration parameters.
        show icmpstats - Displays ICMP statistics.
        show interfaces - Shows interface parameters.
        show ipaddresses - Shows current IP addresses.
        [...]
```

Se observă că se pot afişa la acest pas configuraţiile de pe fiecare interfaţă, serverele DNS configurate, statistici, rute, conexiunile active şi o multitudine de alte informaţii.

În exemplul următor se doreşte stabilirea unei configuraţii statice pe o anumită interfaţă. Pentru aceasta, este necesar să se ruleze întâi o comandă `show` care să listeze interfeţele de reţea împreună cu numerele lor de ordine:

```
        PS C:\Users\Administrator> netsh interface ipv4 show interfaces

        Idx  Met   MTU   State        Name
        ---  ---  -----  -----------  ---------------------
          1   50 4294967295  connected    Loopback Pseudo-Interface 1
         16   40   1500  connected    Wireless Network Connection
         10   30   1500  disconnected  Local Area Connection
         12   20   1500  connected    Local Area Connection 2
         14   20   1500  connected    Local Area Connection 3
         17   40   1500  disconnected  Local Area Connection 4
```

Dacă se doreşte modificarea configuraţiei pentru interfaţa *wireless*, spre exemplu, din rezultatul obţinut mai sus se reţine valoarea câmpului `Idx` din dreptul lui *Wireless Network Connection* (16 în cazul de faţă).

În continuare, pentru a seta configuraţia statică pe interfaţa *wireless*, se rulează comanda următoare. Valoarea folosită pentru parametrul `name` reprezintă indexul interfeţei returnat de comanda `show` precedentă:

```
        netsh interface ipv4 set address name=16 source=static address=192.168.100.75
mask=255.255.255.0 gateway=192.168.100.1
```

Din moment ce prezenţa unui server DNS este importantă pentru funcţionarea oricărei reţele conectate la Internet şi critică într-o configuraţie *Active Directory*, se poate adăuga adresa unui server DNS folosind `netsh` în modul următor, folosind acelaşi index al interfeţei pentru a o identifica:

```
        netsh interface ipv4 add dnsserver name=16 address=192.168.100.40
```

Adăugarea unui server DNS folosind `netsh` permite specificarea unui parametru suplimentar, `index`, care permite numerotarea şi, implicit, introducerea mai multor servere DNS.

Prin intermediul *PowerShell* se poate obţine rapid o listă detaliată a tuturor interfeţelor de reţea instalate în sistem, prin următoarea comandă

```
        Get-WmiObject -Class Win32_NetworkAdapterConfiguration
```

În continuare, folosind `Win32_NetworkAdapterConfiguration` prin WMI şi extinzând comanda anterioară printr-o comandă de formatare a rezultatului, se pot obţine o listă simplificată a configuraţiilor IP ale interfeţelor de reţea şi din *PowerShell*:

```
        PS C:\Users\Administrator> get-wmiobject Win32_NetworkAdapterConfiguration | format-
table IPAddress, Description -autosize

IPAddress                                 Description
---------                                 -----------
[...]
                                          Broadcom 440x 10/100 Integrated Controller
{192.168.1.2, fe80::14c8:f79a:2ed7:74a6}  Dell Wireless 1390 WLAN Mini-Card #3
{192.168.13.1, fe80::9dd5:cc72:ab31:8927} VMware Virtual Ethernet Adapter for VMnet1
{192.168.216.1, fe80::525:fc37:994d:636f} VMware Virtual Ethernet Adapter for VMnet8
                                          Bluetooth LAN Access Server Driver
[...]
```

3-21: Returnarea interfeţelor şi a adreselor IPv4 şi IPv6 corespunzătoare

Pentru o listare doar a adreselor IP, se poate folosi şi varianta următoare a comenzii:

Material cu drept de autor

```
PS C:\> Get-WmiObject -Class Win32_NetworkAdapterConfiguration -Filter IPEnabled=TRUE -
ComputerName . | Select-Object -Property IPAddress

IPAddress
---------
{192.168.1.2, fe80::14c8:f79a:2ed7:74a6}
{192.168.13.1, fe80::9dd5:cc72:ab31:8927}
{192.168.216.1, fe80::525:fc37:994d:636f}
```

Rezultatului obţinut mai sus i s-a aplicat un filtru după proprietatea `IPEnabled` pentru a selecta doar interfeţele care au adrese configurate. Parametrul `ComputerName` urmat de punct (.) reprezintă maşina locală iar trimiterea prin *pipe* (|) comenzii `Select-Object` cu parametrul `Property` setat indică afişarea doar a proprietăţii `IPAddress` a fiecărui obiect returnat.

Din toate comenzile anterioare ce returnează obiecte cu anumite proprităţi se observă că *PowerShell* afişează implicit doar câteva dintre ele. Pentru a afişa toate proprietăţile unui obiect sau, ca în cazul de faţă, toate informaţiile despre conexiunile de reţea, se poate folosi comanda `Select-Object` cu parametrul `Property`, ca şi mai sus, dar specificându-i-se acestuia să afişeze toate proprietăţile printr-un *wildcard* (*):

```
PS C:\Users\Administrator> Get-WmiObject -Class Win32_NetworkAdapterConfiguration -
Filter IPEnabled=TRUE -ComputerName .| Select-Object -Property * -ExcludeProperty IPX*,WINS*
DHCPLeaseExpires            : 20080809144238.000000+120
Index                       : 7
Description                 : Dell Wireless 1390 WLAN Mini-Card #3
DHCPEnabled                 : True
DHCPLeaseObtained           : 20080809134238.000000+120
DHCPServer                  : 192.168.1.1
DNSServerSearchOrder        : {192.168.1.1}
IPAddress                   : {192.168.1.2, fe80::14c8:f79a:2ed7:74a6}
IPEnabled                   : True
DefaultIPGateway            : {192.168.1.1}
InterfaceIndex              : 16
IPSubnet                    : {255.255.255.0, 64}
MACAddress                  : 00:19:7E:11:91:64
[...]
```

3-22: Afişarea tuturor proprietăţilor unei interfeţe

Comanda de mai sus returnează informaţii detaliate despre configuraţia IP, TCP, DNS, rute, parametrii ai protocolului IP şi diferite alte capabilităţi ale interfeţei.

O comandă utilă pentru aflarea adrese IP externe de pe o staţie dintr-o reţea locală care foloseşte o adresă privată[1] este cea de mai jos, ce prelucrează rezultatul unei pagini returnate în urma unei cereri HTML:

```
PS C:\> $webc = New-Object system.net.webclient
PS C:\> $webc.DownloadString("http://checkip.dydns.com") -replace "[^\d\.]"
78.3.73.215
```

3.5.1 Verificarea unei conexiuni

Pe lânga tehnicile de verificare a stării conexiunilor descrise în secţiunea 3.4.3 există o serie de utilitare în linie de comandă, ca `ping`, `tracert`, `pathping` şi `arp` ce mai pot fi folosite pentru a detecta anumite probleme într-o reţea. Toate acestea pot fi folosite atât din *prompt*-ul de comandă cât şi din *PowerShell*.

Utilitarul `ping` verifică doar conectivitatea până la nivelul 3 cu o anumită destinaţie. Comportamentul său este asemănător cu cel din Linux, cu diferenţa că implicit va trimite doar 4 pachete ICMP. Pentru o listă cu parametrii suportaţi se foloseşte comanda sub forma `ping`

[1] De reţinut faptul că o comandă precum ipconfig (sau alte comenzi *PowerShell* ce analizează interfeţele) va afişa adresa configurată pe interfaţă, care poate fi o adresă privată, chiar dacă staţia are conectivitate la Internet prin intermediul unui *gateway*.

Material cu drept de autor

`/?`. Din această listă se observă că pentru a trimite mai multe pachete, comanda acceptă parametrul `-n` urmat de numărul dorit.

```
PS C:\Users\Administrator> ping google.com

Pinging google.com [64.233.167.99] with 32 bytes of data:
Reply from 64.233.167.99: bytes=32 time=162ms TTL=246
Reply from 64.233.167.99: bytes=32 time=163ms TTL=246
Reply from 64.233.167.99: bytes=32 time=163ms TTL=246
Reply from 64.233.167.99: bytes=32 time=173ms TTL=246

Ping statistics for 64.233.167.99:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 162ms, Maximum = 173ms, Average = 165ms
```

3-23: Rezultatul unei comenzi *ping*

Utilitarul `tracert` funcţionează pe acelasi protocol ca şi utilitarul `ping` dar urmăreşte calea până la destinaţie, returnând adresa fiecărui router de pe parcurs. Este util pentru a detecta locul în care se e posibil să fi avut loc o întrerupere în calea de la sursă la destinaţie.

```
PS C:\Users\Administrator> tracert yahoo.com

Tracing route to yahoo.com [206.190.60.37]
over a maximum of 30 hops:

  1     3 ms    38 ms     8 ms  mygateway1.ar7 [192.168.1.1]
  2    30 ms    26 ms    25 ms  172.29.252.46
  3    27 ms    25 ms    25 ms  172.29.32.153
  4    31 ms    39 ms    35 ms  gtr10-gtr11.ip.t-com.hr [195.29.240.97]
  5    41 ms    41 ms    41 ms  mil8-hpt-11.mil.seabone.net [195.22.196.81]
  6   143 ms   143 ms   143 ms  ash2-new11-racc3.new.seabone.net [195.22.216.207]
  7   155 ms   155 ms   156 ms  g2-12-bas2.dce.yahoo.com [206.223.115.2]
  8   157 ms   155 ms   150 ms  ae1-p141.msr1.re1.yahoo.com [216.115.108.19]
  9   158 ms   156 ms   156 ms  ge-9-3.bas-a1.re4.yahoo.com [216.39.49.3]
 10   153 ms   153 ms   154 ms  yahoo.com [206.190.60.37]
```

3-24: Rezultatul unei comenzi *tracert*

Se observă că adresa fiecărui router de pe parcurs este tradusă în numele său conform domeniului DNS căreia îi aparţine. Rezolvarea acestor adrese poate întârzia mult rezultatul lui `tracert`. Pentru a împiedica rezolvarea adreselor în nume se poate folosi parametrul `-d`.

Utilitarul `pathping` se aseamănă cu `tracert`, cu excepţia faptului că scopul său este de a localiza legăturile dintre sursă şi destinaţie care pierd ocazional pachete. `Pathping` trimite `ping`-uri fiecărui router de pe parcurs, până la destinaţie şi calculează numărul de răspunsuri primite de la fiecare, arătând procentajul de pachete pierdute pe fiecare legătură.

```
D:\>pathping -n statia1

Tracing route to statia1 [10.54.1.196]
over a maximum of 30 hops:
  0  172.16.87.35
  1  172.16.87.218
  2  192.168.52.1
  3  192.168.80.1
  4  10.54.247.14
  5  10.54.1.196

Computing statistics for 125 seconds...
            Source to Here   This Node/Link
Hop  RTT    Lost/Sent = Pct  Lost/Sent = Pct  Address
  0                                           172.16.87.35
                                   0/ 100 =  0%   |
  1   41ms     0/ 100 =  0%     0/ 100 =  0%  172.16.87.218
                                  13/ 100 = 13%   |
  2   22ms    16/ 100 = 16%      3/ 100 =  3%  192.168.52.1
                                   0/ 100 =  0%   |
  3   24ms    13/ 100 = 13%      0/ 100 =  0%  192.168.80.1
                                   0/ 100 =  0%   |
  4   21ms    14/ 100 = 14%      1/ 100 =  1%  10.54.247.14
                                   0/ 100 =  0%   |
  5   24ms    13/ 100 = 13%      0/ 100 =  0%  10.54.1.196

Trace complete.
```

Material cu drept de autor

3-25: Rezultatul unei comenzi *pathping*

Atât `ping`, `pathping` cât şi `tracert` sunt utilitare care se bazează pe funcţionarea protocolului de nivel 3 ICMP (*Internet Control Message Protocol*). Totuşi, traficul ICMP este implicit blocat de către *Windows Firewall* atât în Windows *Vista* cât şi în *Windows Server 2008* şi poate fi, de asemenea, blocat de unele routere sau alte *firewall*-uri (*hardware* sau *software*). În consecinţă, o practică utilă înaintea testării conectivităţii prin unul dintre utilitarele enumerate mai sus o reprezintă verificarea permisiunilor pentru protocolul ICMP cel puţin prin *firewall*-urile sursei şi destinaţiei. De notat faptul că excepţia în *Windows Firewall* pentru ICMP se poate adăuga extrem de uşor şi doar prin activarea *file sharing*-ului.

ARP reprezintă atât numele unui utilitar cât şi al unui protocol. Protocolul ARP (*Address Resolution Protocol*) este folosit pentru a menţine asocierea dintre adresele IPv4 din reţeaua locală şi adresele fizice, MAC ale interfeţelor, pentru a se putea realiza adresarea la nivelul 2 în interiorul unui LAN. Utilitarul `arp` oferă posibilitatea de a vizualiza şi a modifica aceste asocieri.

Vizualizarea asocierilor memorate prin protocolul ARP este utilă pentru identificarea unor asocieri incorecte. Spre exemplu, în cazul conectării unor maşini virtuale printr-o reţea locală, utilitarul `arp` poate indica o eroare prin faptul că cele două maşini virtuale folosesc aceeaşi adresă MAC. Tot prin analizarea asocierilor se poate indica şi probabilitatea unui atac de tip *ARP poisoning*, în care atacatorul introduce forţat în tabelele ARP ale staţiilor propria adresă MAC, de regulă asociată cu adresa IP a *gateway*-ului, interceptând astfel traficul destinat în afara reţelei.

Mai jos este exemplificat rezultatul comenzii `arp -a` pentru o memorie ARP supusă unui atac de tip *ARP poisoning*. Se observă că adresele IP 192.168.1.1, 192.168.1.52 şi 192.168.1.53 sunt toate asociate aceleiaşi adrese fizice, 00-19-5b-22-31-a3. Într-un astfel de caz, dacă una dintre adresele 192.168.1.52 sau 192.168.1.53 aparţine atacatorului iar 192.168.1.1 este adresa *gateway*-ului, acesta va intercepta toate pachetele trimise de la staţia curentă în exteriorul reţelei locale (în mod normal, spre *gateway*).

```
PS C:\Users\Administrator> arp -a

Interface: 192.168.1.2 --- 0x10
  Internet Address      Physical Address      Type
  192.168.1.1           00-19-5b-22-31-a3     dynamic
  192.168.1.50          00-18-3a-78-99-f0     dynamic
  192.168.1.52          00-19-5b-22-31-a3     dynamic
  192.168.1.53          00-19-5b-22-31-a3     dynamic
  192.168.1.64          00-1d-60-1c-b5-35     dynamic
[...]
```

3-26: Comanda *arp -a* şi ARP poisoning

Protocolul ARP nu funcţionează pentru IPv6. În cazul lui IPv6, maparea adreselor IP la adrese MAC se face printr-un protocol denumit ND (*Network Discovery*). Drept urmare atacurile de tip *ARP poisoning* nu funcţionează în reţele ce folosesc adresarea prin IPv6.

3.5.2 Configurări statice şi dinamice prin *PowerShell*

Avantajele *PowerShell*-ului stau în primul rând în posibilitatea de a automatiza sarcini complexe sau consumatoare de timp. În practică, o configuraţie statică sau dinamică a parametrilor de reţea, făcută din *PowerShell* nu e cu nimic diferită de una obţinută prin `netsh` sau prin intermediul *Network Connections*, dar pentru reţele mari sau pentru situaţii în care configurarea poate deveni mai complexă sau particulară pentru anumiţi utilizatori sau sisteme, este de dorit utilizarea avantajelor oferite de *PowerShell*, atât din punctul de vedere al complexităţii cât şi din cel al uşurinţei de automatizare şi răspândire a scripturilor.

Material cu drept de autor

Pentru setarea configurațiilor dinamice sau statice asupra interfețelor de rețea se va folosi WMI și clasa `Win32_NetworkAdapterConfiguration` care permite modificarea parametrilor necesari pentru a realiza aceasta.

3.5.2.1 Activarea configurării automate

Pentru a configura interfețele de rețea din sistem să își obțină configurațiile automat, prin DHCP, se poate crea un script prin pașii următori:

1. Clasa `Win32_NetworkAdapterConfiguration` deține o multitudine de metode și proprietăți ce pot fi enumerate prin comanda `Get-Member`. Se observă că deține o metodă numită `EnableDHCP` și o proprietate `IPEnabled` ce vor fi folosite în continuare:

```
Get-WmiObject win32_networkadapterconfiguration | Get-Member
```

2. Pentru a activa configurația prin DHCP, trebuie rulată metoda `EnableDHCP` în contextul fiecărei interfețe de rețea. În consecință, se obține o listă a interfețelor filtrate după cele capabile de configurații IP, pentru a exclude interfețele ce folosesc NetBEUI, IPX/SPX, AppleTalk, etc, și se rulează metoda de mai sus pentru fiecare interfață obținută astfel. Scriptul care realizează aceasta este următorul:

```
$interfete = Get-WmiObject Win32_NetworkAdapterConfiguration | where {$_.IPEnabled -eq "TRUE"}
        foreach($interfata in $interfete) {$interfata.EnableDHCP()}
```

3. În acest moment, dacă se analizează starea configurației din *Network Connections*, se va observa faptul că interfețele sunt, intr-adevăr, configurate pentru a-și obține parametrii IP prin DHCP, dar că serverele DNS sunt în continuare setate la valorile statice. Pentru aceasta, se va folosi metoda `SetDNSServerSearchOrder` cu parametru nul. Metoda primește, de fapt, o listă de servere DNS pe care o va configura pe interfața respectivă în ordinea în care au fost trimise ca parametri. În cazul în care nu primește un parametru ce descrie un server DNS, metoda va activa configurarea automată pentru serverele DNS. Scriptul se completează cu o linie:

```
$interfete = Get-WMIObject Win32_NetworkAdapterConfiguration | where{$_.IPEnabled -eq "TRUE"}
        foreach($interfata in $interfete)
        {
         $interfata.EnableDHCP()
         $interfata.SetDNSServerSearchOrder()
        }
```

4. Se poate folosi acum comanda `ipconfig /all` pentru a verifica noua configurație. Dacă s-a realizat cu succes contactarea serverului DHCP, trebuie să fie vizibil și intervalul de timp pentru care configurația automată este validă (valorile `Lease Obtained` și `Lease Expires`).

3.5.2.2 Activarea configurării statice

Din lista de metode obținută anterior se selectează acele metode ce setează parametrii unei configurații statice și se includ în scriptul următor:

```
$interfete = Get-WMIObject Win32_NetworkAdapterConfiguration | where{$_.IPEnabled -eq "TRUE"}
        foreach($interfata in $interfete) {
          $interfata.EnableStatic("192.168.171.42", "255.255.255.0")
          $interfata.SetGateways("192.168.171.1")
          $DNSServers = "198.102.234.125","198.102.234.126"
          $interfata.SetDNSServerSearchOrder($DNSServers)
          $interfata.SetDynamicDNSRegistration("TRUE")
          $interfata.SetWINSServer("198.102.234.125", "198.102.234.126")
        }
```

Drept rezultat, sunt configurate cu valori statice: adresa IP, masca de rețea, default gateway-ul, servere DNS (principal și secundar) și servere Wins. Configurația completă poate fi verificată de la proprietățile conexiunii, `TCP/IP Properties`, la categoria *Advanced*. Aici pot fi vizualizate configurațiile IP, DNS și WINS.

Din moment ce *PowerShell* lucrează cu tipuri de date bine definite, este important de remarcat tipurile unor parametri folosiți în scriptul de mai sus:

Material cu drept de autor

- Metoda `EnableStatic` primește doi parametri, adresa IP și masca de rețea corespunzătoare;
- Metoda `SetDNSServerSearchOrder` primește un vector de șiruri ce identifică serverele, ce este definit anterior în variabila `$DNSServers`;
- Metoda `SetDynamicDNSRegistration` primește o valoarea booleană și specifică faptul că sistemul va încerca înregistrarea automată a adresei sale de rețea în corespondență cu numele calculatorului (cel definit la *System Properties*, din *Control Panel*), prin DNS. Facilitatea este utilă în cazul în care o stație își schimbă rețeaua din care face parte iar adresarea în interiorul rețelei se face parțial sau preponderent pe bază de nume;
- Metoda `SetWINSServer` primește direct doi parametri, ce reprezintă serverul primar și cel secundar, și nu un vector, ca în cazul serverelor DNS.

3.6 Studiu de caz

O universitate a decis crearea în cadrul campusului a unei rețele informatice cu acces la Internet și a obținut următorul spațiu de adrese: 210.89.32.0/24. Rețeaua urmează a fi împărțită astfel: pentru departamentul de cercetare va exista o subrețea de 13 calculatoare, pentru laboratoare 2 subrețele a câte 25 de calculatoare, pentru secretariat o subrețea de 9 calculatoare iar în cămin va exista o subrețea cu 50 de calculatoare.

1. Care este clasa din care face parte spațiul de adrese primit?
2. Cum trebuie împărțit spațiul de adrese pentru a respecta cerințele problemei?
3. Ce adresa vom aloca *gateway*-ului (ruterului de ieșire din rețea) pentru rețeaua din cămin?
4. Care este adresa de difuzare pentru rețeaua departamentului de cercetare?
5. Câte adrese de stație sunt disponibile în rețeaua utilizată de către secretariat? Care sunt aceste adrese?
6. Care este a 5-a adresă de stație din primul laborator?

Răspunsuri

1. Exprimarea zecimală a primului octet din adresa de rețea disponibilă este cuprins în intervalul [192 – 223], ceea ce înseamnă că primii trei biți ai acestui octet sunt „110", deci adresa primită aparține clasei C.

2. Pentru a rezolva problema trebuie create cele cinci subrețele descrise în enunț. Mai întâi vom încerca să împărțim spațiul disponibil astfel încât toate rețelele să poată conține același număr de stații. Pentru aceasta trebuie să decidem câți biți sunt necesari pentru definirea unei subrețele (astfel încât să putem crea 5 distincte). Răspunsul la această întrebare este dat de cea mai mică putere a lui 2 care generează un număr mai mare sau egal cu necesarul de rețele. În cazul nostru, 2^3=8 și 8 este mai mare decât 5, așadar, pentru a crea 5 subrețele sunt necesari 3 biți. Acești trei biți (de subrețea) se vor adăuga în continuarea măștii de rețea a spațiului ce urmează a fi împărțit, formând noua mască de rețea pentru noile subrețele. Masca de rețea pentru o clasă C este /24, adică este formată din primii 24 de biți consecutivi; adăugând în continuarea acestora încă trei, obținem masca /27 (255.255.255.224).

```
210 .  89 .  32 .  0 0 0  0 0 0 0 0
   24 de biți     3 biți    5 biți
     rețea       subrețea  stație
```

Cele 8 subrețele ce pot fi definite sunt identificate prin biții de subrețea:

```
210. 89. 32.  000 00000   /27    - prima subrețea
210. 89. 32.  001 00000   /27    - a doua subrețea
[...]
```

Numărul de biți rămași disponibili pentru câmpul de stație este 5 (32 – 27). Înseamnă că putem avea câte 2^5-2 = 32-2 = 30 de adrese de stație în fiecare subrețea. Acest lucru nu corespunde însă cu cerințele problemei: rețeaua din cămin trebuie să conțină 50 de stații.

Pentru a satisface cerințele impuse de problemă trebuie folosite măști de rețea cu lungime variabilă.

Material cu drept de autor

Se ia rețeaua cu cel mai mare număr de stații necesar, în cazul de față, rețeaua din cămin, cu 50 de stații, și se va decide câți biți sunt necesari pentru a defini 50 de adrese de stație. Cum nu putem utiliza adresa de rețea (cea în care toți biții de stație sunt 0) și nici adresa de difuzare (cea în care toți biții de stație sunt 1), trebuie găsită cea mai mică putere a lui 2 ce generează un număr cu cel puțin doi mai mare decât numărul necesar. În cazul de față, 2^6-2 = 64 – 2 = 62 și 62 este mai mare decât 50, ceea ce înseamnă că sunt necesari 6 biți pentru asigurarea spațiului de adrese pentru rețeaua din cămin. Prin folosirea a 6 biți din cei 8 disponibili, rezultă că pentru a defini subrețeaua ne mai rămân 2 biți, adică prefixul noi subrețele va fi /26. Adresa de rețea pentru rețeaua din cămin este 210. 89. 32. 01 000000 /26, adică 210.89.32.64 /26.

Au mai rămas de definit rețelele pentru laboratoare, cercetare și secretariat. Numărul cel mai mare de stații este cel pentru rețelele din laborator, 25 de stații. Vom proceda la fel ca mai sus, alegând numărul de biți necesar pentru definirea adreselor de stații. Acesta este 5 (2^5-2 = 30 și 30 > 25). Pentru rețeaua din cămin am folosit una dintre cele 4 rețele /26 pe care le-am creat. Putem împărți o rețea /26 în două rețele /27 prin extinderea măștii de rețea cu un bit. Spre exemplu, a treia rețea /26 creată: 210. 89.32. 10 00000 /26, adică 210. 89.32. 128 /26 are 32-26 = 6 biți disponibili (neacoperiți de masca de rețea). Acești biți pot fi împărțiți la rândul lor (așa cum am făcut cu spațiul inițial) în biți de rețea și biți de stație. Cum noi avem nevoie de 5 biți pentru stații într-o rețea de laborator, mai rămâne un bit pentru subrețea. Cu un bit putem defini două rețele (2^1, deoarece cu un bit putem reprezenta 2 valori). În problema noastră, exact de două rețele a câte 25 de calculatoare este nevoie. Așadar, împărțim a treia rețea /26 în 2 rețele /27 și obținem cele două rețele pentru laboratoare:

```
210.89.32.10000000  /26   =  210.89.32.128   /26
210.89.32.10100000  /26   =  210.89.32.160   /26
```

Rețeaua pentru departamentul de cercetare va conține 13 stații, iar cea pentru secretariat 9. Cea mai mică putere a lui 2 care generează un număr mai mare cu cel puțin 2 decât 13 este 4. La fel și pentru 9. Înseamnă că mai avem de creat 2 rețele cu câte 4 biți pentru definirea adreselor de stație.

Cea de-a patra rețea /26 : 210.89.32. 11 000000, adică 210.89.32.192 are 6 biți disponibili. Cu acești biți trebuie să creăm 2 rețele cu maxim 14 stații. Putem alege să folosim un singur bit pentru a extinde masca de rețea și vom obține 2 rețele iar cu cei 5 biți rămași putem defini câte 2^5 – 2 = 30 de adrese de stație; sau, putem alege să extindem cu 2 biți masca de rețea, pentru a forma 4 subrețele a câte 14 stații. În primul caz rămân adrese de stație nefolosite, în al doilea caz rămân 2 subrețele disponibile pentru alte utilizări. Deși problema nu specifică nici un fel de constrângeri în acest sens, vom alege să folosim 2 biți pentru subrețea, adică să împărțim rețeaua /26 în patru subrețele /28 și să folosim 2 dintre ele pentru departamentul de cercetare și secretariat. Pentru departamentul de cercetare alegem a 2-a subrețea din cele 4 create iar pentru secretariat pe a 3-a. Adresele celor două rețele vor fi:

```
210.89.32.11010000 /28  = 210.89.32.208 /28 - Rețeaua cercetare
210.89.32.11100000 /28  = 210.89.32.224 /28 - Rețeaua secretariat
```

3. În realitate, se poate aloca *gateway*-ului orice adresă validă de stație, există însă o convenție de care bine să ținem cont. Prin convenție, pentru *gateway* se folosește prima adresă validă de stație, adică, în cazul rețelei din cămin, reprezentarea binară a lui 1 pe 6 biți (cinci de 0 urmați de un 1).
   Adresa *gateway*-ului pentru rețeaua din cămin:
   `210.89.32.01000001 /26  =  210.89.32.65 /26`

4. Adresa de rețea pentru rețeaua destinată departamentului de cercetare este:
   `210.89.32.208 /28 = 210.89.32.11010000 /28`
   Adresa de difuzare este cea care are toți biții de stație sunt 1:
   `210.89.32.11011111 /28  adică 210.89.32.223 /28`

Material cu drept de autor

5. Adresa rețelei utilizate de către secretariat este
   210.89.32.224 /28 = 210.89.32.11100000 /28
   Cu cei patru biți din zona de stație putem defini 16 adrese. Întrucât nu putem folosi adresa de rețea și nici adresa de broadcast, rezultă că rămân 14 adrese valide pentru stații. Acestea sunt:
   210.89.32.11100001 /28 – prima adresă de stație
   [...]
   210.89.32.11101110 /28 – ultima adresă de stație
   Cu alte cuvinte, adresele alocabile stațiilor pentru secretariat sunt cele cuprinse în intervalul de adrese 210.89.32.225 – 210.89.32.238

6. Considerăm că primul laborator este cel cu adresa de rețea 210.89.32.128 /26.
   Cea de-a 5-a adresă de stație este:
   210. 89. 32.10000101 /26, adică 210.89.32.133 /26

Material cu drept de autor

Întrebări

1. Care dintre adresele de mai jos reprezintă o adresă validă de stație:

- *150.100.2.8/25*
- *177.1.1.192/26*
- *195.3.15.8/22*
- *200.1.1.63/27*

2. Care dintre adresele de mai jos poate fi o adresă de rețea pentru prefixul /26?

- *209.110.19.64*
- *230.14.3.0*
- *120.4.77.196*
- *89.13.13.26*

3. Fie o rețea locală cu un switch, 3 staţii şi un router pentru conectarea la Internet. Care este numărul maxim de intrări în tabela ARP a unei staţii?

- *2*
- *3*
- *4*
- *mai mare de 4*

4. Din reteaua 192.64.12.0 /24 putem crea:

- *a) 62 de subretele a cate 2 statii?*
- *b) 6 subretele a cate 30 de statii?*
- *c) 8 subretele a cate 32 de statii?*
- *d) 16 subretele cu cate 16 statii?*
- *e) 14 subretele cu cate 14 statii?*

5. Care dintre urmatoarele adrese IP sunt rutabile?

- *a) 142.2.16.79 /28*
- *b) 150.12.180.40 / 29*
- *c) 19.0.27.0 /20*

6. Pentru adresa IP 196.36.44.12 /22

- *a) scrieti adresa de retea si adresa de broadcast*
- *b) care este prima si ultima adresa asignabila din retea?*
- *c) care este adresa urmatoarei retele?*

7. Pentru adresa IP 196.36.44.12

- *a) identificati clasa de adrese din care face parte*
- *b) scrieti adresa de retea si adresa de broadcast*
- *c) care este prima si ultima adresa asignabila din retea?*
- *d) care este adresa urmatoarei retele?*

8. Fie spațiul de adrese 188.88.0.0 /23. Dacă trebuie împărțit în 10 subrețele, care va fi adresa celei de-a 29-a staţii din cea de-a 5-a subrețea?

Material cu drept de autor

4 Rutarea în Internet

"If you asked a hundred people walking down the street...I would bet you that 90 of them, if not 99 of them, would ask, 'What's a router?'"

Stewart Wolpin

Ce se învață din acest capitol?

- Ce este o tabelă de rutare
- Tipuri de rute folosite pentru rutarea traficului
- Protocoale de rutare
- Diferențe între protocoalele de rutare și protocoalele rutate
- Tipuri de protocoale de rutare
- Configurarea rutării în Linux
- Configurarea rutării în Windows

Cine este...

Len Bosack *și* **Sandy Lerner** *sunt co-fondatory Cisco Systems. Bosack și Lerner sunt creatorii primului router multi-protocol, dispozitiv care s-a dovedit un succes comercial. În 1990, Bosack și Lerner au părăsit Cisco și au vândut acțiunile pentru o valoare estimată de 170 de milioane de dolari.*

4.1 Protocoale de rutare și protocoale rutate

4.1.1 Ce este Internetul?

Deși Internetul este un termen întâlnit la tot pasul în domeniul calculatoarelor, definiția sa nu este foarte clară.

Ce este Internetul?

Un răspuns frecvent spune că Internetul este ansamblul tuturor rețelelor interconectate. Problema cu această definiție este că este o definiție descriptivă și, deși adevărată în cele din urmă, este în mare parte irelevantă.

Dacă un calculator nou este conectat la un alt calculator, va fi noul calculator parte din Internet?

Dacă noul calculator are o placă de rețea, atunci va avea și adresă MAC, singura problemă fiind aceea de a face rost și de a atribui calculatorului o adresă IP. O calitate importantă a Internetului constă în capacitatea calculatoarelor de a fi *„TCP/IP compliant"*, adică de a rula stiva de protocoale TCP/IP și de a poseda o adresă IP.

> ***Internetul** poate fi definit mai precis ca ansamblul global al tuturor rețelelor interconectate ce sunt "TCP/IP compliant".*

După cum s-a precizat, pentru a conecta două calculatoare este nevoie de diverse dispozitive de interconectare. Un astfel de dispozitiv este routerul. Routerul face posibilă scalabilitatea Internetului, și astfel chiar existența sa. Prin urmare, orice definiție relevantă a Internetului trebuie să pornească mai degrabă de la routere decât de la stații.

Odată cu dispozitivele de interconectare apar și numeroase protocoale. Privit din punctul de vedere al funcționării sale, Internetul este definit de simbioza a două tipuri de protocoale de nivel rețea: protocoale de rutare și protocoale rutate.

> ***Protocoalele de rutare** sunt cele ce stabilesc regulile prin care informațiile despre rețele sunt schimbate între routere în scopul obținerii unei tabele de rutare adecvate topologiei.*

Material cu drept de autor

Protocoalele rutate *sunt acele protocoale responsabile pentru asigurarea unui mod de identificare a entităţilor ce participă în Internet prin stabilirea unei scheme de adresare ce trebuie să asigure unicitatea şi ierarhizarea adreselor.*

4.1.2 Tabele de rutare

În capitolul Ethernet au fost prezentate tabelele de comutare precum şi procesul de decizie pentru switchuri. Tabela de comutare este o listă de reguli, fiecare cuprinzând o parte de identificare (*matching*) şi una de acţiune, în speţă interfaţa de ieşire din switch (portul). În partea de identificare se află o adresă MAC destinaţie, iar în partea de acţiune este precizată una din interfeţele switchului.

Datorită dimensiunii mult mai mari a Internetului acest mod de decizie a trebuit rafinat în două direcţii: alegerea unei scheme de adresare ierarhice, pe de o parte, şi implementarea unor algoritmi cât mai performanţi de căutare şi implicit de stocare a informaţiilor, pe de altă parte.

În ceea ce priveşte prima direcţie, deşi are o vârstă venerabilă, IPv4 oferă o schemă de adresare satisfăcătoare, deşi puţin flexibilă. Prin urmare, IPv6 va trebui să ofere multe alte avantaje pe lângă extinderea lungimii adresei de la 32 la 128 de biţi, pentru a reuşi să se impună drept înlocuitorul lui IPv4. Cea de-a doua direcţie este îndeplinită cu ajutorul unor algoritmi de caching specifici dispozitivului de rutare şi sistemului de operare asociat.

Similar cu switchul, routerul foloseşte o tabelă de rutare pentru dirijarea pachetelor catre destinaţie. O intrare în tabela de rutare se mai numeşte şi rută.

O ***rută*** *este o regulă ce cuprinde o parte de identificare şi una de acţiune. Partea de identificare este compusă din două elemente: adresa reţelei destinaţie şi masca acesteia, în vreme ce partea de acţiune poate fi exprimată prin ambele sau doar unul dintre următoarele elemente: adresa următorului router (numită next hop address) şi interfaţa de ieşire din router.*

O ***tabelă de rutare*** *este o listă de rute cu acces secvenţial.*

În figura de mai jos este prezentată o tabelă de rutare. Important de remarcat este faptul că, deşi folosirea tabelei de rutare se face analizând secvenţial rutele începând cu prima, construcţia tabelei se face prin inserarea unei noi rute în faţa primei rute cu un prefix mai scurt decât aceasta. Drept rezultat lungimea măştii de reţea va scădea odată cu parcurgerea tabelei de rutare. Altfel spus, informaţiile referitoare la reţelele mici se vor găsi înaintea informaţiilor despre reţelele mai mari.

Se mai poate observa că pentru unele rute este precizată doar adresa următorului router, în vreme ce pentru altele doar interfaţa de ieşire. Deşi adresa următorului hop este întotdeauna de ajuns pentru specificarea completă a unei rute, informaţia despre interfaţa de ieşire este insuficientă în cazul în care această interfaţă este conectată la un mediu multiacces. Astfel, deşi o rută validă poate preciza doar interfaţa de ieşire în cazul unei legături seriale (sau orice altă legătură punct la punct), aceeaşi rută este considerată ambiguă în cazul unei legături Ethernet, în acest al doilea caz fiind necesară precizarea adresei următorului hop.

Adresă reţea	Mască	Next hop	Interfaţă
194.230.85.0	/26	172.17.0.9	E0
194.230.85.128	/26	-	S0
194.230.85.0	/24	194.230.5.65	E1
194.230.86.0	/24	199.17.17.0	-

4-1: Tabelă de rutare cu patru intrări

Material cu drept de autor

Pentru a înțelege mai uşor modul de decizie a unui router se va presupune că un router cu o tabelă de rutare identică cu cea prezentată în figura de mai sus primeşte un pachet cu adresa destinație 194.230.85.66.

Din întregul pachet singura informație relevantă pentru nivelul rețea al unui router este adresa logică destinație. În primul rând, routerul va trebui să determine dacă nu este el destinatarul acestui pachet, iar pentru aceasta va compara adresele logice ale tuturor interfețelor sale active cu adresa destinație a pachetului. Dacă este destinatarul pachetului atunci va trimite datele nivelurilor superioare.

În cazul în care routerul nu are nicio interfață activă cu aceeaşi adresă ca cea a pachetului, routerul va trece la pasul doi, încercând să determine dacă destinația se află în aceeaşi rețea ca şi sursa. Pentru aceasta va analiza adresa şi masca interfeței pe care a primit pachetul în cauză. Astfel va aplica masca asociată interfeței de intrare pe adresa interfeței urmând ca rezultatul să fie comparat cu rezultatul operației de „şi" logic între aceeaşi mască de rețea şi adresa destinație. Dacă cele două rezultate coincid, routerul va ignora pachetul, altfel urmând să înceapă procesul efectiv de rutare.

Prima rută din tabela de rutare va fi extrasă, iar rezultatul operației de „şi" logic între adresa destinație şi masca de rețea cuprinsă în această rută va fi comparat cu adresa de rețea. În cazul în care cele două valori coincid, antetul de nivel legătură de date al pachetului va fi rescris (antetul de nivel rețea rămânând neschimbat) şi pachetul va fi trimis către următorul hop. Dacă valorile diferă, va fi extrasă următoarea rută până la găsirea primei potriviri sau până la epuizarea rutelor, caz în care pachetul urmează să fie ignorat.

În cazul exemplului de față, se presupune că pachetul cu destinația 194.230.85.66 trebuie rutat. Prima rută va aplica masca /26 adresei destinație, rezultând 194.230.85.64, valoare diferită de 194.230.85.0. Nici a doua rută nu este potrivită pentru această destinație, astfel ajungându-se la cea de a treia rută.

Aparent prima şi a treia rută se referă la aceeaşi rețea. La o privire mai atentă măştile de rețea diferă, astfel a treia rută se referă la un spațiu de adrese de patru ori mai mare decât prima. În realitate, datorită caracterului secvențial al utilizării tabelelor de rutare şi a faptului că primele două rute se referă la două sferturi din acelaşi spațiu de adrese, numărul de adrese diferite ce vor folosi a treia rută va fi doar dublu (şi nu de patru ori mai mare) decât numărul de adrese ce vor folosi prima rută.

În urma aplicării măştii /24 pentru adresa destinație, rezultă 194.230.85.0 şi anume adresa rețelei destinație din această rută. Pentru a putea trimite pachetul va trebui aflată adresa MAC a următorului hop, şi anume 194.230.5.65. În urma interogării tabelei ARP sau a unei cereri ARP, adresa MAC sursă este rescrisă cu adresa MAC a interfeței E1, iar adresa MAC destinație este înlocuită cu adresa MAC corespunzătoare următorului hop.

4.1.3 Clasificarea rutelor

Există numeroase criterii de clasificare a rutelor. Unele criterii prezintă o relevanță redusă pentru Internetul secolului 21. Câteva dintre ele vor fi amintite totuşi, deoarece, în anii de expansiune a Internetului, structurau înțelegerea rețelei globale.

La începutul anilor '90 o rută nu conținea masca de rețea, întregul proces de rutare fiind classfull. Astfel, o primă clasificare ar fi în funcție de tipul procesului de rutare, şi anume **rute classfull** sau **rute classless**. Odată cu trecerea timpului şi cu creşterea în popularitate a adresării classless tabelele de rutare au devenit classless chiar dacă sunt alimentate uneori de protocoale de rutare classfull (adică protocoale ce nu transmit şi informații despre masca de rețea), routerele urmând să precizeze explicit masca de rețea înainte de a introduce informațiile în tabela de rutare.

Material cu drept de autor

În rutarea cu rute classfull procesul este diferit, şi anume adresa destinaţie extrasă din antetul unui pachet ajuns la router va fi mai întâi comparată cu 192, şi în cazul în care e mai mică de 192 va fi comparată cu 128, determinându-se astfel clasa de adrese şi implicit masca de reţea. Din acest punct procesul este similar cu cel din rutarea *classless*, adică se va efectua o operaţie de „şi" logic între adresa destinaţie şi masca reţelei, rezultatul urmând a fi comparat cu adresa de reţea conţinută în rută.

Odată cu răspândirea rutării classless a apărut clasificarea rutelor în funcţie de tipul destinaţiei. Astfel sunt **rute de tip nod** (sau rute *host*) şi **rute de tip reţea**. Rutele *host* conţin informaţii doar despre o singură staţie, adică masca de reţea este /32. Odată cu creşterea Internetului, şi implicit a dimensiunii tabelelor de rutare, a crescut şi presiunea de a agrega cât mai mult rutele, precum şi a se renunţa la rutele de tip nod. Cu toate acestea, datorită modului de inserare a rutelor, şi deci datorită promovării rutelor de tip nod la începutul tabelei de rutare, acestea având prefixul maxim, rutele *host* mai sunt încă folosite pentru unele optimizări de trafic, mai ales pe routerele de la periferia Internetului.

Al treilea criteriu de clasificare a rutelor este în funcţie de modul de conectare, iar cele două tipuri de rute sunt: **rutele direct conectate** şi **rutele gateway (către reţele distante).** Rutele direct conectate sunt rute către reţele în care routerul în cauză are o interfaţă, şi în majoritatea cazurilor aceste rute sunt automat introduse în tabela de rutare de sistemul de operare în momentul configurării şi activării interfeţei respective. Rutele direct conectate nu conţin, în general, adresa următorului hop, având specificată doar interfaţa de ieşire din router. Astfel, rutele direct conectate sunt singurele rute valide care pot avea specificată ca interfaţă de ieşire o interfaţă multiaccess (gen *Ethernet*), fără a necesita precizarea adresei următorului hop.

O a patra clasificare a rutelor se face în funcţie de mediul de acces la reţea, având astfel **rute pe medii punct la punct** şi **rute pe medii multiacces**. Rutele către o reţea conectată pe o legătură punct la punct pot fi specificate ori prin interfaţa de ieşire din router, ori prin adresa următorului hop, ori prin ambele. Rutele pe medii multiacces sunt specificate doar prin adresa următorului hop, interfaţa de ieşire nefiind suficientă.

Din punctul de vedere al unui router, două medii de transmisie acoperă marea majoritate a rutelor; altfel spus, două tipuri anume de interfeţe sunt mult mai populare decât restul. Cele două tipuri de interfeţe sunt interfeţele Ethernet şi cele seriale. Ethernet permite transmisia peste un mediu multiacces, în vreme ce interfaţa serială este punct la punct. Alte interfeţe punct la punct destul de răspândite sunt cele de fibră optică şi cele ISDN.

Un tip special de rută este ruta implicită.

***O ruta implicită (rută default)** este ruta spre care se trimit toate pachetele pentru care nu se cunoaşte o destinaţie specifică.*

Altfel spus, ruta implicită este ruta care se potriveşte cu toate destinaţiile, având în partea de adresă de reţea din rută un spaţiu de adrese ce cuprinde toate adresele IP. Acest spaţiu de adrese este 0.0.0.0/0 şi, deşi deseori ruta implicită este denumită ca ruta cu 4 de zero sau quad-zero route, caracteristica distinctivă a acestei rute se află în masca de lungime zero.

Există situaţii în care pot exista mai multe rute implicite într-o tabelă de rutare. În tabela de rutare de mai jos, ultimele două rute sunt două rute implicite.

Adresă reţea	Mască	Next hop	Interfaţă
.....			
0.0.0.0	/0	194.230.5.65	S0
0.0.0.0	/0	-	S1

4-2: Tabelă de rutare cu două rute implicite

Material cu drept de autor

Dacă ambele rute apar în tabela de rutare, este evident că niciun pachet nu va ajunge să fie prelucrat de cea de-a doua rută implicită, toate pachetele fiind acceptate de prima.

Dezactivarea unei interfețe, ca urmare a închiderii administrative sau a întreruperii legăturii de nivel fizic sau de nivel legătură de date, are ca rezultat înlăturarea tuturor rutelor ce folosesc respectiva interfață ca interfață de ieșire din router. Astfel, în absența celei de-a doua rute implicite în cazul dezactivării interfeței S0, toate pachetele care ar fi fost rutate prin prima rută implicită ar urma să fie ignorate.

În concluzie, într-o tabelă de rutare există o singură rută implicită activă, dar pot fi precizate mai multe rute *default* în scopuri de *backup*.

Ultima clasificare este și cea mai relevantă în prezent. Pornind de la informația pe baza căreia sunt construite rutele, distingem **rutele statice** de **rutele dinamice**. Această clasificare se referă doar la rutele gateway, deoarece rutele direct conectate sunt introduse automat în tabela de rutare.

4.1.4 Rute statice

Rutele statice sunt introduse manual de administratorul routerului, spre deosebire de rutele dinamice care necesită configurarea unui protocol de rutare. O rută statică va apărea în tabela de rutare numai atunci când interfața de ieșire din router este configurată corect și activată.

O limitare a folosirii rutelor statice este că schimbările în topologia rețelei trebuie cel mai adesea să fie actualizate manual pe router de administrator. Deși există modalități de minimizare a efectului schimbărilor în topologie, aceste procedee duc la mărirea dimensiunii tabelei de rutare.

Pe de altă parte, orice protocol de rutare necesită o anumită lățime de bandă pentru actualizarea tabelelor de rutare. Odată primită, informația de rutare trebuie prelucrată înainte de a fi introdusă în tabela de rutare, unele protocoale de rutare consumând cantități semnificative de timp de procesor sau de memorie. Spre deosebire de rutarea dinamică, rutarea statică nu are cerințe de lățime de bandă, timp de procesor sau memorie (în afară de memoria efectivă ocupată de tabela de rutare). Introducerea manuală de informații poate părea o metodă arhaică de administrare, dar, în cazul routerelor de la periferia Internetului, rutarea statică este adesea mai potrivită decât rutarea dinamică.

Multe dintre rețelele de mici dimensiuni nu au o legătură redundantă la Internet, astfel încât specificarea legăturii de Internet printr-o rută statică sau dinamică sunt la fel de inutile în cazul întreruperii conexiunii.

În concluzie, rutele statice nu sunt o soluție depășită de configurare a routerelor, iar viitorul nu pare să anunțe înlocuirea rutării statice cu rutarea dinamică. Rutarea statică continuă să fie soluția optimă pentru rețelele cu un grad redus de complexitate sau pentru rețelele *stub* (rețele cu o singură conexiune la Internet).

4.2 Protocoale rutate

Protocolul IP, prezentat într-un capitol anterior, este de fapt singurul protocol rutat folosit în Internet începând cu anii 2000. Cele două protocoale rutate concurente pentru IP sunt Apple Talk și IPX. Implementările soluțiilor de conectare peste Internet bazate pe aceste protocoale sunt rare, dar cele două protocoale rămân încă destul de folosite în rețelele locale.

Material cu drept de autor

Deşi Macintosh ocupă 3,5% din piaţa mondială de calculatoare personale, efortul pentru susţinerea soluţiilor bazate pe Apple Talk în rândul marilor producători de echipamente de reţea s-a diminuat rapid, trecând pe un plan colateral începând cu anii `95.

IPX, în schimb, a fost multă vreme un competitor redutabil pentru IP. De fapt, competiţia dintre stiva de protocoale TCP/IP şi SPX/IPX s-a tradus în limbajul reţelelor locale de calculatoare în competiţia dintre Microsoft şi Novell.

O discuţie a avantajelor unuia dintre protocoale faţă de celălalt poate ocupa destul spaţiu. În cele din urmă, răspunsul la întrebarea „de ce IP, şi nu IPX?" ţine mai degrabă de raţiuni de marketing decât de raţiuni tehnice.

Singura concurenţă reală pentru protocolul IPv4 este protocolul IPv6, dar slaba sa răspândire actuală poate susţine afirmaţia că există, de fapt, un singur protocol rutat în Internet. Mai mult, migraţia către IPv6 nu a început din nucleul Internetului pentru a se propaga apoi spre reţelele locale. În prezent există reţele locale IPv6 legate cel mai adesea prin mecanisme de tunelare peste IPv4.

4.3 Protocoale de rutare

4.3.1 Determinarea căii optime

Protocoalele de rutare, uneori denumite şi protocoale de rutare dinamică, au drept obiectiv schimbarea informaţiilor despre reţelele cunoscute între routerele ce rulează acelaşi protocol de rutare. Pe baza acestor informaţii se construiesc rutele dinamice.

Routerele au câte o tabelă de rutare pentru fiecare protocol rutat. Pentru un protocol rutat dat informaţiile oferite de toate protocoalele de rutare se regăsesc într-o singură tabelă. Prin urmare, aceeaşi tabelă de rutare va conţine rutele direct conectate, rutele statice şi cele dinamice.

Un router poate rula unul sau mai multe protocoale de rutare, numărul protocoalelor de rutare ce pot fi rulate fiind limitat în general de sistemul de operare sau de modelul routerului. Un router Cisco, de exemplu, rulează în general până la 30 de instanţe de protocoale de rutare.

O problemă care apare este că acelaşi protocol de rutare poate să furnizeze două sau mai multe rute către aceeaşi destinaţie; pot, de asemenea, exista două rute dinamice către aceeaşi reţea provenite din protocoale de rutare diferite.

Astfel, deşi există trei tipuri de rute, este necesară specificarea un mecanism de comparare a rutelor între ele. Mai mult, trebuie ca toate rutele să fie ierarhizate. Cele două criterii de ierarhizare a rutelor sunt distanţa administrativă şi metrica.

***Distanţa administrativă** este un număr între 0 şi 255, asociat cu un tip de rută sau cu un protocol de rutare, ce permite ierarhizarea protocoalelor de rutare.*

***Metrica** unei rute este un număr, rezultat din aprecierea calităţii unui drum spre o anumită destinaţie în raport cu un set de criterii. Metrica şi setul de criterii sunt relevante pentru un anumit protocol de rutare. Prin urmare, nu are sens compararea metricilor unor rute obţinute prin protocoale de rutare diferite.*

Altfel spus, distanţa administrativă deosebeşte două protocoale de rutare, în vreme ce metrica deosebeşte două rute ale aceluiaşi protocol de rutare. Rutele statice sunt singurele rute pentru care distanţa administrativă poate fi schimbată.

Distanţele administrative pentru unele dintre cele mai folosite protocoale de rutare sunt precizate în tabelul de mai jos:

Material cu drept de autor

Tip rută	Distanța administrativă
Direct conectată	0
Rută statică	1
Rută agregată	5
BGP	20
EIGRP	90
IGRP	100
OSPF	110
IS-IS	115
RIP	120
ODR	160
iBGP	200

4-3: Valorile distanțelor administrative

4.3.2 Clasificarea protocoalelor de rutare

Există numeroase clasificări ale protocoalelor de rutare. În continuare vor fi prezentate două dintre acestea.

În vreme ce stațiile sunt grupate în rețele pentru a oferi o ierarhizare a spațiului de adrese, rețelele sunt grupate, la rândul lor, în colecții de rețele aflate sub o administrație comună numite sisteme autonome (AS). Astfel, o primă clasificare a protocoalelor de rutare se face în **protocoale de rutare inter-AS**, folosite pentru schimbul informațiilor de rutare între sisteme autonome diferite și **protocoale de rutare internă,** adică protocoale folosite în cadrul aceluiași sistem autonom.

Cea de-a doua clasificare a protocoalelor de rutare se referă doar la clasificarea protocoalelor de rutare internă, în funcție de modul de schimbare a informației de rutare. Cele trei clase în care sunt împărțite protocoalele de rutare internă sunt: **protocoale bazate pe vectori de distanță** (*distance-vector protocols*), **protocoale bazate pe starea conexiunii** (*link-state protocols*) și **protocoale hibride**.

Cea de a treia categorie de protocoale de rutare internă îmbină caracteristici ale protocoalelor bazate pe vectori de distanță cu unele caracteristici ale protocoalelor bazate pe starea conexiunii. Această nouă categorie este folosită pentru a clasifica protocoalele ce nu pot face parte din niciuna dintre primele două categorii, astfel că, în cele ce urmează, vor fi prezentate doar protocoalele *distance-vector* și *link-state*.

4.3.3 Protocoale distance-vector

Pentru protocoalele de tip *distance-vector* calculul drumului optim se face pe bază de direcție (indicată de obicei prin precizarea interfeței) și distanța până la destinație folosind direcția respectivă. Informațiile de rutare se schimbă numai între routerele învecinate, la intervale periodice. Aceasta rezultă într-un timp de convergență mare, adică schimbările apărute în topologia rețelei se propagă destul de greu către rutele mai îndepărtate de locul apariției modificării în cauză.

Cele mai populare protocoale de rutare bazate pe vectori de distanță sunt RIP[1] și IGRP, acestea fiind ușor de configurat și consumând resurse de lățime de bandă și timp de procesor foarte reduse.

[1] http://tools.ietf.org/html/rfc1058

Material cu drept de autor

RIP foloseşte drept metrică numărul de hopuri sau routere până la reţeaua destinaţie. Pentru a se evita efectele negative ale buclelor logice a fost stabilită o metrică maximă, astfel încât orice informaţie despre o rută cu o metrică mai mare de 15 este ignorată.

Actualizările se fac transmiţând toate informaţiile de rutare şi nu doar cele ce s-au modificat de la ultima actualizare, dar sunt trimise folosindu-se adrese de difuzare. Prin urmare, pachetele de actualizare vor ajunge doar la routerele adiacente, deoarece în mod implicit routerele filtrează pachetele de *broadcast*.

Actualizările se fac implicit la 30 de secunde, acest timp reprezentând un compromis între timpul de convergenţă şi cantitatea de bandă utilizată pentru actualizări. Astfel, timpul de convergenţă la RIP în cel mai defavorabil caz este de 7 minute jumătate (15 hopuri), calificând RIP în categoria protocoalelor de rutare internă cu o convergenţă scăzută. În măsura în care se impune un timp de convergenţă mai mic, perioada de actualizare poate fi redusă, ducând însă la un consum mai ridicat de lăţime de bandă.

Fiecare router ce primeşte un pachet de actualizare va incrementa metrica fiecărei rute conţinute în pachet cu 1, deoarece rutele conţinute în pachet îi sunt accesibile prin intermediul unui hop suplimentar, şi anume routerul ce a trimis pachetul de actualizare. Apoi, pentru fiecare dintre rutele din pachetul de actualizare, va încerca să determine dacă nu există deja o rută cu o metrică mai bună către aceeaşi destinaţie în tabela de rutare.

4.3.4 Protocoale link state

Protocoalele de tip *link-state* (bazate pe starea conexiunii) construiesc o bază de date cu întreaga topologie a reţelei şi calculează drumul cel mai scurt pe baza unui algoritm de tip Dijkstra (SPF - *shortest path first*).

Astfel, pentru actualizarea tabelelor de rutare se trimite într-o primă etapă întreaga tabelă de rutare către toate routerele ce rulează acelaşi protocol de rutare, aceasta realizându-se prin folosirea în câmpul destinaţie a unei adrese logice de multicast specifice fiecărui protocol în parte. După această etapă de trimitere a tuturor informaţiilor, numită şi *flooding*, actualizările se vor efectua doar la apariţia unei schimbări în topologie, iar pachetele de actualizare vor conţine doar informaţii despre rutele modificate, acestă metodă de actualizare numindu-se actualizare incrementală.

Principala problemă a acestor tipuri de protocoale este că fiecare dintre routere va trebui să construiască arborele topologic, şi apoi să extragă rutele ca drumuri optime în acest arbore. Acest proces necesită resurse de memorie şi procesor semnificative. În plus, efortul configurării unui protocol bazat pe starea conexiunii este adesea mult mai mare decât cel necesar pentru a configura un protocol bazat pe vectori de distanţă.

Cu toate acestea timpul de convergenţă pentru protocoalele *link-state* este semnificativ mai redus decât pentru protocoalele *distance-vector*. Aceasta se datorează iniţierii procesului de actualizare odată cu apariţia modificărilor în topologie, precum şi folosirii adresării multicast, şi deci a propagării informaţiilor de actualizare în întreaga reţea.

4.4 Sisteme autonome

4.4.1 Ce este un sistem autonom?

Dată fiind dimensiunea Internetului, toate protocoalele rutate trebuie să suporte o schemă de adresare ierarhică. Cu toate acestea, la ora actuală există zeci de milioane de reţele, astfel încât un router din nucleul Internetului ar avea o tabelă de rutare uriaşă.

În realitate, în plus faţă de cele două niveluri de ierarhizare aduse de protocolul rutat, reţelele sunt grupate în sisteme autonome sau AS (*autonomous systems*).

Material cu drept de autor

*Un **sistem autonom** este o colecție de routere aflate sub o administrație comună.*

O administrație comună se referă la un set comun de protocoale de rutare, un set de politici de securitate și de criterii de decizie. O caracteristică importantă a unui AS este faptul că orice ISP, pentru a putea activa în Internet, trebuie să se afilieze unui sistem autonom.

Un sistem autonom este identificat printr-un **număr AS** denumit și **adresă AS**. Acest număr poate fi cuprins între 1 și 65.535, cu toate că ultimul segment al acestui spațiu de adrese, și anume numerele între 64.512 și 65.535, sunt rezervate pentru uz privat, similar claselor de adrese IP private.

Atribuirea unui număr AS se face de IANA (*Internet Assigned Numbers Authority*). Prețul de 100 USD este nesemnificativ, dar costul real pentru un ISP ține de birocrația obținerii adresei. Procedura de justificare a unui număr AS face imposibilă obținerea unui număr AS de ISP-urile mici și medii. Singura opțiune pentru acestea este de a intra într-un AS deja existent și de a accepta regulile și politicile de securitate și rutare ale acestuia.

4.4.2 Protocoale de rutare inter-AS

Odată cu gruparea rețelelor în sisteme autonome a apărut și problema dezvoltării de protocoale care să facă față cerințelor rutării între AS-uri. În acest scop s-a definit clasificarea protocoalelor în protocoale de tip IGP (*Interior Gateway Protocol*) și protocoale de tip EGP (*Exterior Gateway Protocol*). IGP sunt protocoalele de rutare interioare unui sistem autonom, iar EGP sunt protocoalele de rutare exterioare unui AS, sau, altfel spus, protocoale de rutare inter-AS.

Prima și cea mai importantă cerință pentru un protocol de tip EGP este de a face față unor tabele de rutare semnificativ mai mari decât orice se poate întâlni în interiorul unui AS. O tabelă de Internet actuală care este schimbată între două routere de graniță din sisteme autonome diferite cuprinde aproximativ 180.000 de rute.

A doua cerință este cea de flexibilitate. Deși unele dintre protocoalele interne folosesc până la cinci factori pentru determinarea metricii, folosirea unei formule pentru determinarea costului unei rute nu oferă un grad suficient de flexibilitate. Procesul complet de comparare a două sau mai multe rute pentru BGPv4, spre exemplu, este un algoritm cu 13 pași.

În plus, numărul informațiilor asociate cu o rută crește semnificativ. Pentru un IGP singurele informații transmise sunt: rețea destinație, mască, metrică, în vreme ce pentru BGP mai trebuie precizate și atribute ca: `origin`, `as_path`, `next_hop`, `local_pref`, `atomic_aggregate`, `aggregator`, `multi_exit_disc`.

O altă schimbare față de rutarea internă o reprezintă schimbarea paradigmei de securitate. Astfel, dacă s-ar considera un protocol de rutare inter-AS după criteriile rutării clasice (adică interne), EGP ar fi în categoria extrem-paranoic. Această schimbare a paradigmei de securitate, corelată cu cerința de flexibilitate, se traduce printr-o creștere a gradului de complexitate a configurării unui EGP.

Pe de altă parte, cerințele de convergență pentru un EGP sunt destul de reduse, datorită faptului că legăturile de nucleu sunt extrem de stabile. Astfel, timpul de convergență pentru BGP este de ordinul orelor mai degrabă decât al minutelor - precum în cazul unui protocol de rutare internă.

Spre deosebire de rutarea internă, rutarea inter-AS nu lasă loc pentru existența mai multor protocoale diferite, datorită cantității importante de resurse, dar și datorită numărului relativ redus de sisteme autonome. Câștigătorul competiției este, fără îndoială, BGPv4[1].

[1] http://tools.ietf.org/html/rfc4271

Material cu drept de autor

În principiu, BGP funcționează la fel ca orice protocol de rutare, în sensul că schimbă tabele de rutare cu routerele vecine. În plus față de un IGP, BGP mai transmite odată cu acestea și informația de AS, precum și o serie de alte atribute. Astfel, un router BGP va construi așa-numitele **AS-paths**, specificând AS-urile prin care trebuie sa treacă un pachet până la destinație. Folosind aceste AS-paths, BGP evită buclele de rutare.

Este important de precizat că BGP transmite mesajele încapsulate în segmente TCP, folosind portul dedicat 179. Drept consecință, înainte de a putea rula o sesiune BGP, trebuie ca între cele două noduri să existe rute. Cel mai adesea alimentarea inițială a tabelelor de rutare este realizată de un IGP, dar la fel de bine conexiunea poate fi stabilită și prin precizarea unor rute statice.

Înainte de a încheia capitolul trebuie accentuat că nu există o ierarhie unică a protocoalelor de rutare. Nu există un „cel mai bun protocol de rutare" care să fie optim pentru orice rețea. Deseori, insistând pe comparația dintre diferite protocoale, se scapă din vedere că soluția cea mai bună pentru un context dat poate fi rutarea statică. Acest lucru este valabil nu doar în rețelele de mici dimensiuni, ci chiar și pentru interconectarea a două sisteme autonome diferite. BGP oferă un grad ridicat de flexibilitate, dar pentru a exploata această flexibilitate este nevoie de mai mult de o conexiune către Internet. Pentru un singur furnizor de servicii Internet, o singură rută implicită poate fi suficientă.

4.5 Configurări la nivel de router în Linux

Sistemul de operare Linux permite configurarea unei stații ca router. Aceasta înseamnă că stația respectivă poate să primească pachete IP de la alte stații din rețea, să determine calea spre destinația pachetelor și, în cazul în care este posibil, să trimită traficul către destinație.

Pentru a realiza aceste funcții, se menține la nivelul nucleului o tabelă de rutare, pe baza căreia se determină traseul către destinația unui pachet.

O observație importantă referitoare la rutarea din Linux este că nucleul lucrează simultan cu două tabele de rutare: *FIB (Forwarding Information Base)* este tabela uzuală la care se face referire prin termenul de tabelă de rutare, iar *routing cache* este o tabelă ce accelerează rutarea prin păstrarea ultimelor rute utilizate într-o memorie cache.

După cum s-a prezentat în secțiunile anterioare, există două tipuri de rute: statice (adăugate manual de utilizatorul sistemului) și dinamice (învățate prin intermediul unui protocol de rutare, precum RIP, OSPF, BGP). Datorită faptului că, în general, routerele Linux se afla la periferia Internetului, rutele statice sunt suficiente pentru a asigura rutarea.

4.5.1 Activarea rutării

Pentru a permite comutarea pachetelor între două interfețe ale aceleiași mașini (activarea comportamentului de router), trebuie activată variabila `ip_forward`. Activarea se poate realiza temporar sau permanent.

Activarea temporară se realizează prin intermediul `procfs`:

```
cuirass:~# cat /proc/sys/net/ipv4/ip_forward
0
cuirass:~# echo 1 > /proc/sys/net/ipv4/ip_forward
cuirass:~# cat /proc/sys/net/ipv4/ip_forward
1
```

sau `sysctl`:

```
cuirass:~# sysctl -a | grep ip_forward
net.ipv4.ip_forward = 0
cuirass:~# sysctl -w 'net.ipv4.ip_forward=1'
net.ipv4.ip_forward = 1
cuirass:~# sysctl -a | grep ip_forward
net.ipv4.ip_forward = 1
```

Material cu drept de autor

Configurarea permanentă se realizează prin editarea fișierului `sysctl.conf`:

```
       cuirass:~# cat /etc/sysctl.conf | grep -B 1 ip_forward
# Uncomment the next line to enable packet forwarding for IPv4
#net.ipv4.ip_forward=1
cuirass:~# vi /etc/sysctl.conf
cuirass:~# cat /etc/sysctl.conf | grep -B 1 ip_forward
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

Fișierul va fi interpretat în momentul repornirii sistemului. Se poate forța repornirea sa tot prin intermediul interfeței `sysctl`:

```
       cuirass:~# sysctl -a | grep ip_forward
net.ipv4.ip_forward = 0
cuirass:~# sysctl -p
net.ipv4.ip_forward = 1
cuirass:~# sysctl -a | grep ip_forward
net.ipv4.ip_forward = 1
```

4.5.2 Configurarea rutelor

În mod tradițional, utilitarul `route` amintit mai sus este folosite pentru a adăuga, respectiv pentru a șterge rute din tabela de rutare.

Începând de la versiunea de nucleu 2.2 a apărut un nou utilitar pentru configurarea parametrilor de rețea, cunoscut sub numele **iproute**. Ajuns la cea de a doua versiune (`iproute2`), acesta reprezintă o alternativă puternică, permițând realizarea de configurări foarte sofisticate. Utilitarul `iproute` este folosit prin intermediul comenzii `ip`. Documentația utilitarului este dată de pagina de manual (`man 8 ip`). În cazul rutării o documentație minimală poate fi vizualizată prin rularea comenzii:

```
       cuirass:~# ip r help
Usage: ip route { list | flush } SELECTOR
       ip route get ADDRESS [ from ADDRESS iif STRING ]
                            [ oif STRING ]  [ tos TOS ]
       ip route { add | del | change | append | replace | monitor } ROUTE
       [...]
```

Exemplele prezentate în continuare vor folosi preponderent utilitarul `iproute2`.

4.5.2.1 Vizualizare tabelei de rutare

Tabela de rutare se afișează prin invocarea comenzii ip cu argumentul `route`:

```
       cuirass:~# ip r
172.16.68.0/24 dev eth0  proto kernel  scope link  src 172.16.68.128
default via 172.16.68.2 dev eth0
```

Se observă că se pot reduce argumentele la o secvență minimă unică. Se pot selecta intrările din tabela de rutare după anumite criterii, folosind argumentul `list`:

```
       cuirass:~# ip r l type local
cuirass:~# ip r l type unicast
172.16.68.0/24 dev eth0  proto kernel  scope link  src 172.16.68.128
default via 172.16.68.2 dev eth0
       cuirass:~# ip r l scope link
172.16.68.0/24 dev eth0  proto kernel  src 172.16.68.128
cuirass:~# ip r l scope host
cuirass:~# ip r l scope global
default via 172.16.68.2 dev eth0
```

În tabelul de mai sus există o rută direct conectată (*link*) și o rută implicită. Ruta direct conectată este interfața `eth0` prin care sistemul se conectează la rețeaua locală. Ruta implicită folosește ca *gateway* 172.16.68.2.

4.5.2.2 Adăugarea de rute

Pentru a adăuga o rută către o rețea / *host*, se folosește comanda `ip r a`:

```
       cuirass:~# ip r l
```

Material cu drept de autor

```
       172.16.68.0/24 dev eth0  proto kernel  scope link  src 172.16.68.128
default via 172.16.68.2 dev eth0
cuirass:~# ip r a 192.168.38.100/32 via 172.16.68.2
cuirass:~# ip r a 192.168.38.0/24 via 172.16.68.2
cuirass:~# ip r l
192.168.38.100 via 172.16.68.2 dev eth0
192.168.38.0/24 via 172.16.68.2 dev eth0
172.16.68.0/24 dev eth0  proto kernel  scope link  src 172.16.68.128
default via 172.16.68.4 dev eth0
```

În exemplul de mai sus s-a adăugat o rută de tip *host* către stația cu adresa IP 192.168.38.100 și o rută de tip rețea către rețeaua 192.168.38.0.

În exemplul următor se adaugă o rută implicită:

```
       cuirass:~# ip r a default via 172.16.68.4
cuirass:~# ip r l
192.168.38.100 via 172.16.68.2 dev eth0
192.168.38.0/24 via 172.16.68.2 dev eth0
172.16.68.0/24 dev eth0  proto kernel  scope link  src 172.16.68.128
default via 172.16.68.2 dev eth0
```

Tabela de rutare poate fi alterată prin intermediul comenzii `ip r change`:

```
       cuirass:~# ip r c default via 172.16.68.5
cuirass:~# ip r l
192.168.38.100 via 172.16.68.2 dev eth0
192.168.38.0/24 via 172.16.68.2 dev eth0
172.16.68.0/24 dev eth0  proto kernel  scope link  src 172.16.68.128
default via 172.16.68.5 dev eth0
```

Sunt întâlnite frecvent cazuri în care este nevoie de mai multe rute către o singură destinație. Motivul principal al unei astfel de implementări este *redundanța*. În cazul în care o rută devine inutilizabilă, poate fi folosită una dintre rutele de rezervă. Criteriul pentru stabilirea rutei principale este costul acesteia. Din punct de vedere al sistemului de operare, prioritatea rutelor este stabilită de *metrică*. Rutele instalate folosind comenzile de mai sus au metrica default 0. Pentru schimbarea metricii (și prioritizarea rutelor) se poate folosi opțiunea `metric` la adăugarea unei intrări în tabela de rutare:

```
       cuirass:~# ip r a 192.168.3.2 via 172.16.68.5 metric 5
cuirass:~# ip r l match 192.168.3.2
192.168.3.2 via 172.16.68.5 dev eth0  metric 5
default via 172.16.68.5 dev eth0
```

Comanda `ip r l match` este folosită pentru a afișa intrările din tabela de rutare care corespund unei anumite rețele. În exemplul de mai sus se afișează ruta de tip *host* către 192.168.3.2 și ruta implicită care se potrivește cu orice adresă.

4.5.2.3 Ștergerea de rute

Ștergerea de rute este mai simplă decât adăugarea, deoarece nu trebuie specificată decât rețeaua / stația spre care duce ruta.

Rutele din tabela de rutare pot fi șterse cu ajutorul comenzii `ip route del`. În exemplele de mai jos se elimină o rută de tip *host*, o rută de tip rețea și o rută implicită:

```
       cuirass:~# ip r l
192.168.3.2 via 172.16.68.5 dev eth0  metric 5
192.168.38.100 via 172.16.68.2 dev eth0
192.168.38.0/24 via 172.16.68.2 dev eth0
172.16.68.0/24 dev eth0  proto kernel  scope link  src 172.16.68.128
default via 172.16.68.5 dev eth0
cuirass:~# ip r d 192.168.3.2
cuirass:~# ip r d 192.168.38.0/24
cuirass:~# ip r d default
cuirass:~# ip r l
192.168.38.100 via 172.16.68.2 dev eth0
172.16.68.0/24 dev eth0  proto kernel  scope link  src 172.16.68.128
```

Material cu drept de autor

4.5.2.4 Adăugarea de rute cu caracter permanent

Pentru ca modificările efectuate asupra tabelei de rutare să aibă caracter permanent, se foloseşte fişierul `/etc/network/interfaces`. În acest fişier pot fi configurate comenzi care să adauge rute la pornirea sistemului sau în momentul folosirii scripturilor de interacţiune specifice (`/etc/init.d/networking`).

Configurarea rutei implicite se realizează prin intermediul directivei *gateway*. Mai jos este precizată o configuraţie care adaugă ruta implicită către 192.168.1.1 şi o rută către staţia 192.168.38.100:

```
        cuirass:~# cat /etc/network/interfaces
[...]
        # The primary network interface
allow-hotplug eth0
auto eth0
iface eth0 inet static
        address 172.16.68.128
        netmask 255.255.255.0
        broadcast 172.16.68.255
        gateway 172.16.68.2
        dns-nameservers 172.16.68.2
        up ip route add 192.168.38.100/32 via 172.16.68.5
        cuirass:~# /etc/init.d/networking restart
Reconfiguring network interfaces...done.
cuirass:~# ip r l
192.168.38.100 via 172.16.68.5 dev eth0
172.16.68.0/24 dev eth0  proto kernel  scope link  src 172.16.68.128
default via 172.16.68.2 dev eth0
```

Opţiunea up este folosită la rularea unei comenzi după activarea interfeţei. În exemplul de mai sus, după activarea interfeţei se adaugă o rută către reţeaua 192.168.38.100.

4.6 NAT - Network Address Translation

NAT[1] oferă posibilitatea schimbării unei adrese IP cu o alta din antetul unui pachet IP. În practică, NAT se foloseşte pentru a permite staţiilor care utilizează adrese private să acceseze Internetul.

Deoarece există un număr destul de mare de reţele neconectate la Internet, IETF a încercat să reglementeze folosirea adreselor în cadrul acestor reţele. Soluţia[2] a fost definirea unor spaţii de adrese private, adrese ce nu pot fi rutate pe Internet.

Clasa	Spaţiul de adrese	
A	10.0.0.0/8	10.0.0.0 – 10.255.255.255
B	172.16.0.0/12	172.16.0.0 – 172.31.255.255
C	192.168.0.0/16	192.168.0.0 – 192.168.255.255

4-4: Clasele private de adrese

Obţinerea unei reţele are un cost fix de 50$ plus TVA. Costul este foarte mic, dar procesul este anevoios, deoarece necesită justificarea dimensiunii spaţiului de adrese, precum şi o aşteptare de până la 3 luni.

Folosirea adreselor private oferă o schemă de adresare rapidă şi comodă. În plus, datorită faptului că adresele staţiilor nu sunt accesibile din afara reţelei, folosirea adreselor private este deseori considerată una dintre cele mai eficiente politici de securitate.

În acelaşi timp, adresele private pun o serie de probleme. Cea mai importantă este faptul că routerul prin care reţeaua privată accesează Internetul va trebui să fie capabil să facă conversia adreselor private în adrese publice, deci să ruleze un serviciu de NAT. Acest serviciu

[1] http://tools.ietf.org/html/rfc1631
[2] http://tools.ietf.org/html/rfc1918

Material cu drept de autor

impune o latență suplimentară pentru fiecare pachet ce tranzitează routerul. Un alt dezavantaj este acela că în interiorul unei rețele private nu se recomandă plasarea de calculatoare care oferă servicii publice, întrucât este dificil să fie inițiate conexiuni din exterior către acestea.

Translatarea statică a adreselor presupune constituirea unei tabele de translatare ce va atribui întotdeauna aceeași adresă publică pentru o adresă privată dată. Trebuie să existe în acest caz un număr egal de adrese publice și adrese private.

Translatarea dinamică a adreselor presupune definirea unui rezervor de adrese publice, care apoi vor fi atribuite în funcție de ordinea în care stațiile solicită conexiuni cu Internetul. Odată încheiată ultima conexiune a unei stații, adresa publică este returnată în rezervorul de adrese, putând fi alocată unei alte stații. Avantajul major al acestei implementări este că numărul adreselor publice poate fi semnificativ mai redus decât al celor private. Dacă spre exemplu se ia o rețea cu 200 de calculatoare, dar din care pe Internet nu sunt niciodată mai mult de un sfert, atunci se poate realiza o translatare dinamică a adreselor pe un spațiu de 64 de adrese publice, și nu 256 ca în cazul translatării statice.

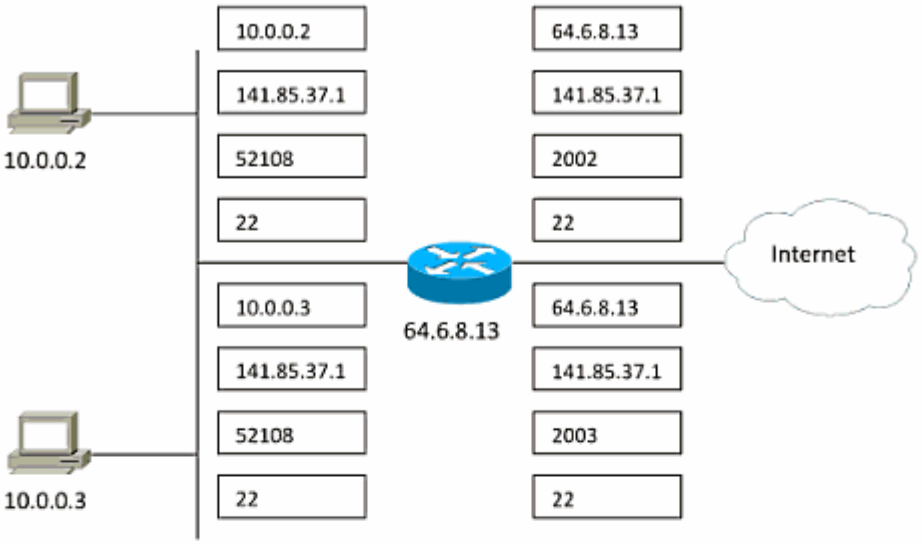
Translatarea adreselor cu supraîncărcare este forma cea mai des întâlnită de NAT. În prezent, când se face referire la translatarea adreselor, în realitate se face referire la translatarea adreselor cu supraîncărcare. Această metodă oferă posibilitatea folosirii unei singure adrese publice pentru mai multe stații ce accesează Internetul. Din această cauză această metodă mai este numită și NAT multi-la-unu sau PAT (*Port Address Translation*).

Avantajul PAT este că permite un număr de aproximativ 64000 de conversații simultane de la orice *host* intern către exterior cu o singură adresă externă. Implementarea înlocuiește pachetul din rețeaua locală cu adresa sursă S, adresa destinație D, portul sursă P, portul destinație Q, cu altul nou ce va avea adresa sursă M (adresa routerului), adresa destinație D, portul sursă K. Portul destinație nu se schimbă. De asemenea se memorează asocierea (S,P) - K. Dacă un pachet ajunge pe router din exterior, având adresa destinație M, adresa sursă Q și portul destinație K, atunci acest pachet va fi înlocuit cu un altul cu adresa destinație S, adresa sursă Q, portul destinație P și va fi trimis în rețeaua locală. Portul sursă nu se schimbă.

Figura de mai jos prezintă un exemplu de PAT. Stațiile din rețeaua locală cu adresele 10.0.0.2 respectiv 10.0.0.3 se conectează la stația 141.85.37.1 prin SSH (portul destinație este 22). Gateway-ul cu adresa 64.6.8.13 ascunde identitatea celor două stații și realizează maparea portului 52108 al stației 10.0.0.2 peste portul 2002 local și cel al portului 52108 al stației 10.0.0.3 peste portul local 2003.

Un caz special al PAT îl reprezintă **redirectarea**. În acest caz se va înlocui pachetul primit din rețeaua locală având adresa sursă S, adresa destinație D, portul P cu un altul având adresa sursă S, adresa destinație M (adresa routerului), portul R (portul în care se face redirectarea, specificat de utilizator). Redirectarea este în general folosită pentru a implementa un *proxy* transparent, caz în care pe routerul M portul R ascultă un *proxy* configurat pentru *proxy* transparent.

Material cu drept de autor



4-5: NAT overloaded (PAT)

4.6.1 Translatarea de adrese în Linux

În Linux, translatarea adreselor se realizează folosind utilitarul `iptables`. Utilitarul `iptables` este folosit pentru NAT şi pentru filtrarea pachetelor (*firewall software*). După cum îi spune şi numele, utilitarul dispune de tabele asociate anumitor scopuri:

- tabela *nat* este folosită pentru NAT;
- tabela *filter* este folosită pentru filtrarea pachetelor;
- tabela *mangle* este o tabelă folosită pentru alterarea avansată a pachetelor.

Fiecare tabelă conţine un set de lanţuri dedicate unui anumit tip de acţiune. În fiecare lanţ pot fi adăugate reguli specifice care definesc modul în care vor fi prelucrate diversele pachete.

Pentru translatarea de adrese se foloseşte **tabela nat**. În această tabelă există trei lanţuri predefinite: `PREROUTING` - modifică pachetul imediat ce acesta intră în router, înainte de a fi rutat, `OUTPUT` - modifică pachetele generate local înainte ca acestea să intre în procesul de rutare, şi `POSTROUTING` - modifică pachetele ce urmează să plece din router, după ce acestea au fost rutate. Ţintele valide sunt `ACCEPT`, `DROP`, `QUEUE`, `REJECT`, `LOG`, `SNAT`, `DNAT`, `MASQUARADE`, `REDIRECT`.

4.6.1.1 Acţiuni în cadrul tabelei nat

Acţiunile posibile în cadrul tabelei NAT sunt SNAT, DNAT, MASQUERADE şi REDIRECT.

`SNAT` se foloseşte pentru a indica o translatare de adrese de tip `PAT` pe adresa sursă. Adresa sursă a pachetului va fi modificată la una din intervalul specificat prin opţiunea `--to-source`. Cu aceeaşi opţiune se poate specifica şi intervalul în care se va alege portul sursă când se face translatarea de adrese. Această ţintă este validă numai în lanţul `POSTROUTING` (şi lanţurile chemate din acest lanţ).

`DNAT` se foloseşte pentru o translatare de adrese de tip `PAT` pe adresa destinaţie. Adresa destinaţie a pachetului va fi modificată la una din intervalul specificat prin opţiunea `--to-destination`. Această ţintă este validă numai în lanţurile `PREROUTING` şi `OUTPUT` (şi lanţurile chemate din acest lanţ).

Material cu drept de autor

MASQUERADE este echivalent cu SNAT. Adresa sursă va fi înlocuită cu adresa setată a interfeței pe care va fi trimis pachetul. Trebuie folosită în loc de SNAT dacă adresa la care se face translatarea este setată dinamic (prin DHCP de exemplu).

REDIRECT se foloseşte pentru a redirecta pachetul, local, pe portul specificat de opțiunea --to-port. Această țintă este validă numai în lanțurile PREROUTING şi OUTPUT.

4.6.1.2 Exemple de utilizare a tabelei nat

Pentru a evidenția mai bine lucrurile menționate mai sus se pot urmări regulile de mai jos:

```
        iptables -t nat -A POSTROUTING -o eth1 -s 192.168.0.0/24 -j SNAT --to-source 1.2.3.4
        iptables -t nat -A PREROUTING -i eth0 -d 14.15.16.17 -j DNAT --to-destination
192.168.100.1
```

Prima regulă poate fi interpretată în felul următor: toate pachetele ce vin cu adresa IP sursă din rețeaua 192.168.0.0/24 vor fi trimise pe interfața eth1 cu adresa IP sursă 1.2.3.4, după ce acestea vor fi rutate. Cea de-a doua regulă va schimba adresa IP destinație (14.15.16.17) a pachetelor ce intră pe interfața eth0 cu adresa IP 192.168.100.1, înainte ca acestea să fie rutate.

În exemplul de mai jos a fost prezentată configurarea iptables pentru translatarea de adrese pe sistemul 141.85.37.1 din rețeaua 192.168.1.0/24. Această maşină este doar router, iar serverul de web, serverul de e-mail şi serverul de DNS rulează pe servere diferite, în rețeaua internă. Routerul are conectată interfața eth0 la rețeaua internă şi interfața eth1 la Internet.

```
        iptables -t nat -A POSTROUTING -o eth1 -s 192.168.1.0/24 -j SNAT --to-source 141.85.37.1
        iptables -t nat -A PREROUTING -i eth1 -p tcp --destination-port 80 -j DNAT --to-
destination 192.168.1.2
        iptables -t nat -A PREROUTING -i eth1 -p tcp --destination-port 25 -j DNAT --to-
destination 192.168.1.3
        iptables -t nat -A PREROUTING -i eth1 -p tcp --destination-port 53 -j DNAT --to-
destination 192.168.1.4
        iptables -t nat -A PREROUTING -i eth1 -p udp --destination-port 53 -j DNAT --to-
destination 192.168.1.4
```

Prima linie din exemplu este folosită pentru ca stațiile din rețea să poată accesa Internetul. La trecerea pachetelor prin router, adresa sursă a stațiilor (adresă privată) va fi înlocuită cu adresa routerului (adresă publică).

Următoarele linii vor trimite traficul de web, e-mail şi DNS către serverele din interior. În acest context, din exterior, aparent routerul este şi server de web, e-mail şi DNS.

Ce se întâmplă dacă se doreşte ca serviciile din rețeaua internă, care sunt făcute publice, să fie văzute din exterior ca rulând pe porturi diferite față de cele standard? Metoda ce poate fi aplicată aici se numeşte **port forwarding**. Această metodă permite unei stații (*firewall*) să trimită cererile ce îi sunt adresate către o altă stație ce va procesa aceste cereri. Cea mai folosită întrebuințare a acestei metode apare când serverele rulează pe stații aflate în rețeaua internă (după *firewall*). Cu alte cuvinte, se presupune existența unui *gateway* cu două interfețe, eth0 fiind conectată la rețeaua internă şi eth1 la Internet. Fie 141.85.37.1 adresa IP publică a interfeței eth1 şi 192.168.0.2 adresa IP a stației pe care rulează un serviciu web, pe portul implicit 80, stație ce va putea fi accesată din exterior. În exemplul de mai jos se va face redirectarea conexiunilor ce vin pe 141.85.37.1:8888 (<IP_extern:port>) către 192.168.0.2:80 (IP_intern:port).

```
        iptables -t nat -A PREROUTING -i eth1 -p tcp -d 141.85.37.1 -dport 8888 -j DNAT --to-
destination 192.168.0.2:80
```

4.6.2 Alterarea avansată a pachetelor

Tabela mangle este folosită pentru a modifica atribute specifice ale pachetelor. NAT modifică doar adresele dintr-un pachet. Tabela *mangle* poate fi folosită pentru a schimba

Material cu drept de autor

informaţii precum TTL, TOS, sau pentru a marca un pachet. Marcarea pachetelor este folosită doar în interiorul routerului. Odată ce un pachet părăseşte routerul, informaţiile adăugate la marcare vor fi îndepărtate. În prezent, marcarea pachetelor este folosită de către sistemul de *Quality of Service* (QoS).

Există trei lanţuri predefinite: `INPUT` - modifică pachetele destinate routerului, `FORWARD` - modifică pachetele în curs de rutare, şi `POSTROUTING` - modifică pachetele după rutare. Ţintele valide includ `ACCEPT`, `DROP`, `QUEUE`, `REJECT`, `LOG` precum şi cele specifice tabelei, prezentate mai jos.

`MARK` marchează pachetul cu valoarea specificată prin opţiunea `--set-mark`. Pachetele marcate se pot folosi ulterior în procesul de rutare sau QoS.

`TOS` setează câmpul de *type of service* la valoarea specificată prin opţiunea `--set-tos`.

`TTL` setează câmpul TTL la valoarea specificată prin opţiunea `--ttl-set`, decrementează valoarea acestuia (dacă se foloseşte opţiunea `--ttl-dec`) sau incrementează valoarea acestuia (dacă se foloseşte opţiunea `--ttl-inc`).

Mai jos este prezentat un exemplu de alterare avansată a pachetelor, în care două staţii din reţeaua internă, `141.85.37.13` şi `141.85.37.169` pot accesa doar reţeaua internă (reţeaua departamentului) şi reţeaua imediat următoare (reţeaua organizaţiei).

```
iptables -t mangle -A FORWARD -s 141.85.37.13 -j TTL --ttl-set 2
iptables -t mangle -A FORWARD -s 141.85.37.169 -j TTL --ttl-set 2
```

4.6.3 Tunelare

O reţea privată virtuală (VPN – *Virtual Private Network*) reprezintă o modalitate de asigurare a unei conexiuni sigure peste o infrastructură publică. Principalele tehnologii ce permit stabilirea unei astfel de conexiuni sunt criptarea şi tunelarea. Odată cu apariţia reţelelor private virtuale de nivel 2, componenta de criptare a fost abandonată, singura caracteristică universală a soluţiilor de VPN rămânând tunelarea.

Principalul scop al tunelării într-o reţea virtuală este de a ascunde (fără a altera) atât identitatea sursei cât şi a destinaţiei faţă de routerele de pe parcurs, routere ce aparţin infrastructurii publice. Dacă s-ar apela la o translatare de adresă, informaţiile originale nu ar fi accesibile decât serverului de translatare de adresă.

În cazul tunelării antetul iniţial al pachetului este păstrat nealterat, ataşându-se un nou antet ce va avea ca adresă sursă adresa capătului local al tunelului, iar ca adresă destinaţie adresa celuilalt capăt al tunelului.

Cel mai adesea tunelarea se realizează la nivelul reţea prin tunelarea traficului IPv4 printr-un tunel IPv4, precum în cazul VPN, sau tunelarea unui alt protocol rutat, spre exemplu tunelarea traficului IPv6 peste o infrastructură IPv4 (tunel *6to4*).

Se poate folosi tunelare şi în cazul traficului de nivel legătură de date, cea mai întâlnită metodă de tunelare fiind tunelarea *dot1q*, folosită de ISP în cazul în care acesta oferă conservarea informaţiilor de VLAN din reţelele clienţilor.

O categorie aparte de tunelare o reprezintă tunelarea de nivel 2,5 folosită spre exemplu în arhitecturile de IP MPLS.

Având în vedere că principalul scop al tunelării IP peste IP este acela de a oferi securitate, trebuie să răspundem la întrebarea:

În ce măsură se obţine un plus de securitate folosind tunelarea, atâta vreme cât informaţiile originale se află în continuare în cadrul original?

Răspunsul la această întrebare ţine de aprecierea volumului foarte mare de date ce trebuie comutate în nucleul Internetului. Valoarea foarte ridicată a traficului determină ca orice prelucrare adiţională a cadrelor să aibă consecinţe semnificative pentru scalabilitatea reţelei. Altfel spus, deşi teoretic este posibilă selectarea foarte precisă doar a pachetelor de

Material cu drept de autor

autentificare pentru sesiunile de telnet spre exemplu, o inspectare atât de amănunțită a tuturor cadrelor ce sunt comutate de router poate duce la o scădere a performanței unui router dedicat de până la o valoare de doar 20% din perfomanța inițială.

Cu toate acestea, în cazul unei infrastructuri publice cu un grad scăzut de siguranță securitatea unei rețele virtuale private nu mai poate fi asigurată doar prin tunelare, fiind necesară și criptarea traficului.

Care este consecința creșterii lungimii pachetelor prin adăugarea unui nou antet în cazul tunelării?

Atâta vreme cât pachetele prin adăugarea noului antet se încadrează în dimensiunea maximă a cadrului (1500 octeți), adăugarea noului antet de tunelare are un impact minor pentru performanțele rețelei.

Dacă, în schimb, se va ajunge la tunelarea pachetelor care după încapsularea de nivel rețea au deja 1500 de octeți, prin adăugarea unui nou antet se va depăși dimensiunea maximă impusă de protocolul de nivel legătură de date, în acest caz urmând să apară o fragmentare în două cadre. Astfel fiecare cadru care în urma tunelării va depăși 1500 de octeți va fi împărțit în două cadre. Chiar și în cazul în care există un trafic susținut între aceași sursă și destinație, trafic bazat pe trasferul de cadre de lungime maximă, în urma tunelării se va dubla numărul de pachete. Deși jumătate din pachete vor fi de dimensiune minimă, acestea vor fi transmise individual, deoarece complexitatea concatenării lor în unul sau mai multe cadre de lungime maximă este considerată prea mare.

În implementările uzuale se dorește nu numai evitarea concatenării fragmentelor de lungime minimă, dar chiar evitarea fragmentării. Acest lucru se realizează prin impunerea unei dimensiuni maxime datelor înainte de încapsularea de nivel rețea.

***MTU** (Maximum Transfer Unit) reprezintă dimensiunea maximă a pachetelor după încapsularea de nivel rețea.*

***MSS** (Maximum Segment Size) reprezintă dimensiunea maximă a pachetelor înainte de încapsularea de nivel rețea.*

În cazul trasferului traficului IPv4 peste un tunel IPv4, se definește MSS ca fiind cu 40 de octeți mai mic decât MTU, acești 40 de octeți reprezentând cele două antete IPv4 ce vor fi adăugate la nivelul rețea.

Pentru a evita fragmentarea, limitarea MSS trebuie operată pe stația sursă. Pentru aceasta în sistemele Windows trebuie modificate două chei în baza de date de regiștri, iar în Linux limitarea se impune prin configurarea `iptables`.

4.6.4 Configurarea tunelului GRE în Linux

În secțiunea 4.6.3 s-a prezentat conceptul de tunelare și se recomandă parcurgerea acesteia înaintea secțiunii următoare.

GRE[1] (*Generic Routing Encapsulation*) este un protocol de tunelare care încapsulează diferite informații în pachete IP. Linux permite crearea de tunele GRE cu ajutorul modulului `ip_gre`:

```
        valhalla:/home/razvan# lsmod | grep ip_gre
ip_gre                 15620  0
```

Crearea tunelelor GRE se realizează cu ajutorul utilitarului `iproute2` prin intermediul comenzii `ip tunnel`.

[1] http://tools.ietf.org/html/rfc1701

Material cu drept de autor

Folosirea tunelelor GRE în Linux va fi exemplificată printr-un scenariu tipic. Există două rețele A și B interconectate printr-o rețea C. Rețelele A și B dețin stații cu adrese private. Routerul R1 este folosit pe post de *gateway* de rețeaua A, iar routerul R2 este folosit pe post de gateway de rețeaua B.

Informațiile asociate celor două rețele sunt prezentate în tabelul de mai jos:

Rețea A		**Rețea B**	
Adresă rețea	172.16.68.0/24	Adresă rețea	10.38.0.0/16
Adresă internă R1	172.16.68.1	Adresă internă R1	10.38.0.1
Adresă externă R1	141.85.37.178	Adresă externă R1	141.85.37.1
Adresă sistem de test	172.16.68.128	Adresă sistem de test	10.38.6.123

Pentru rețeaua A se creează tunelul `netb` și interfața cu același nume:

```
valhalla:~# ip t add netb mode gre remote 141.85.37.1 local 141.85.37.178 ttl 255
valhalla:~# ip l set netb up
valhalla:~# ip addr add 172.16.68.1 dev netb
valhalla:~# ip route add 10.38.0.0/16 dev netb
```

Ruta către rețeaua B este intermediată de interfața virtuală `netb` asociată tunelului GRE.

La fel, pentru rețeaua B se creează tunelul neta și interfața cu același nume:

```
root@csr:~# ip t a neta mode gre remote 141.85.37.178 local 141.85.37.1 ttl 255
root@csr:~# ip l set neta up
root@csr:~# ip a a 10.38.0.1 dev neta
root@csr:~# ip r a 172.16.68.0/24 dev neta
```

În acest moment, deși aflate în rețele private, stațiile pot comunica unele cu celelalte. Prin tunelul folosit toate pachetele de la stațiile din rețeaua A vor ajunge la stațiile din rețeaua B. O captură de pachete folosind `tcpdump` are următorul conținut:

```
       22:07:36.819507 IP 141.85.37.178 > 141.85.37.1: GREv0, length 88: IP 172.16.68.128 >
10.38.6.123: ICMP echo request, id 41995, seq 1, length 64

       22:07:36.820140 IP 141.85.37.1 > 141.85.37.178: GREv0, length 88: IP 10.38.6.123 >
172.16.68.128: ICMP echo reply, id 41995, seq 1, length 64

       22:07:37.827154 IP 141.85.37.178 > 141.85.37.1: GREv0, length 88: IP 172.16.68.128 >
10.38.6.123: ICMP echo request, id 41995, seq 2, length 64

       22:07:37.827657 IP 141.85.37.1 > 141.85.37.178: GREv0, length 88: IP 10.38.6.123 >
172.16.68.128: ICMP echo reply, id 41995, seq 2, length 64
```

Se observă ca stația cu adresa 172.16.68.128 din rețeaua A transmite pachete IMCP Echo request către stația cu adresa 10.38.6.123 din rețeaua B, iar aceasta îi răspunde. Toată comunicația este încapsulată în tunelul GRE. Pachetele ICMP sunt încapsulate în pachete IP în care adresele IP sursă, respectiv destinate sunt adresele IP externe ale routerelor R1 și R2.

4.7 Rutarea în Windows Server 2008

4.7.1 Routing and remote access services

Dispozitivul care reunește două segmente separate de rețea (denumite și subrețele, în contextul rețelelor TCP/IP) este denumit, generic, un router. Principalul rol al unui router este de a decide dacă pachetele care sosesc la el trebuie să ajungă sau nu într-un alt segment de rețea conectat la acesta.

Material cu drept de autor

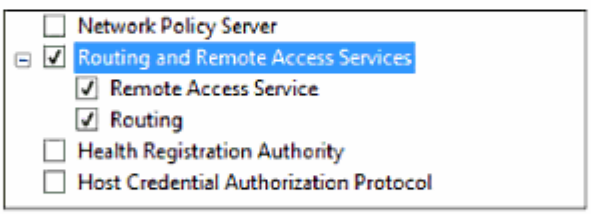
Majoritatea rețelelor folosesc echipamente dedicate în acest scop, al rutării. Acestea reprezintă soluții *hardware* ce funcționează cu propriile sisteme de operare (Cisco, Juniper, etc) și sunt destinate rețelelor de mărime medie și mare.

Windows Server 2008 oferă o soluție *software* pentru rutare prin intermediul serviciilor grupate sub denumirea de *Routing and Remote Access*. Din punct de vedere *hardware*, singurele modificări necesare pentru a implementa un proces de rutare folosind *Windows Server 2008* sunt una sau mai multe plăci de rețea suplimentare.

Soluțiile *software* pentru rutare reprezintă o alternativă viabilă pentru cazurile în care nu este necesară implementarea unor soluții dedicate: izolarea a mici segmente de rețea sau folosirea serverului ca *gateway* spre Internet pentru o rețea locală de mici dimensiuni. În cazul în care *Windows Server 2008* este instalat pe un sistem cu rol de gateway, există opțiunea de a rula și serviciul de NAT (*Network Address Translation*) pentru a oferi conectivitate în exterior unei rețele care folosește adrese private printr-o singură adresă publică.

Principiul de funcționare a procedeului de rutare în *Windows Server 2008* este același ca și în cazul echipamentelor dedicate: deciziile sunt luate pe baza rutelor din tabela de rutare. În plus, serviciul de *Routing and Remote Access* poate fi configurat atât pentru rute statice cât și pentru rute dinamice, obținute prin intermediul unui protocol de rutare. Intrucât *Windows Server 2008* este conceput ca o soluție de rutare pentru rețele de dimensiune mică, protocolul de rutare suportat este RIP (*Routing Internet Protocol*).

Pentru a putea folosi serviciul de rutare în *Windows Server 2008* este necesară adăugarea acestuia ca rol al serverului, *din Server Manager*. Pentru aceasta, de la linkul *Add roles* se selectează *Network Policy and Access Services*. În continuare, înainte de instalare, se oferă opțiunea de a selecta subcomponente ale acestui rol. Pentru rutare este necesară selectarea *Routing and Remote Access Services*, cu ambele sale componente: *Remote Access Service* și *Routing* (figura 4-6).



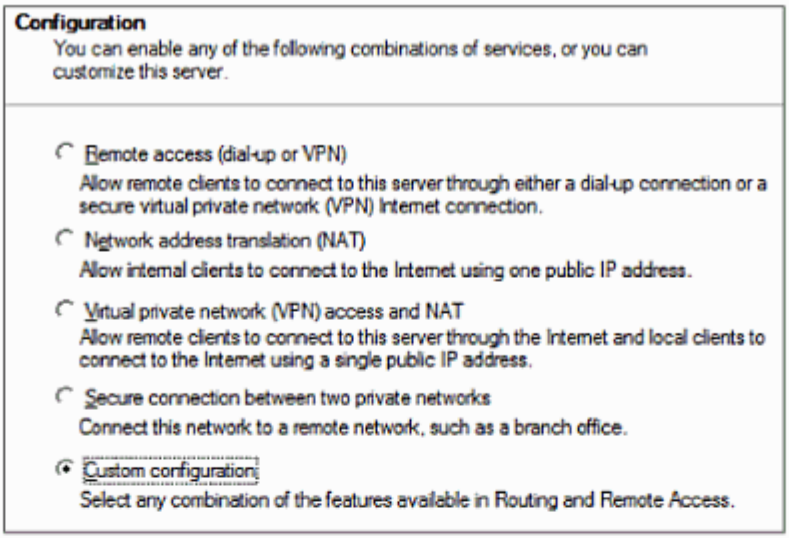
4-6: Instalarea serviciului de Routing and Remote Access Services

Pentru a putea configura rutarea prin RRAS trebuie activat întâi serviciul. Acestea pot fi realizate prin consola *Server Manager*. După instalare, *Routing and Remote Access* poate fi găsit sub *Network Policy and Access Services* în cadrul rolurilor serverului.

Configurarea inițială și pornirea serviciului se face prin selectarea *Routing and Remote Access* și accesarea meniului *More Actions* din panoul de acțiuni urmată de alegerea opțiunii *Configure and Enable Routing and Remote Access*.

***Atenție**, nu se poate activa sau configura serviciul de rutare pe* Windows Server 2008 *atât timp cât mașina pe care este instalat serviciul realizează partajarea unei conexiuni la Internet. Înainte de a porni configurarea conform descrierii de mai sus, trebuie dezactivat ICS (Internet Connection Sharing) pentru toate conexiunile, din interfața Network Connections.*

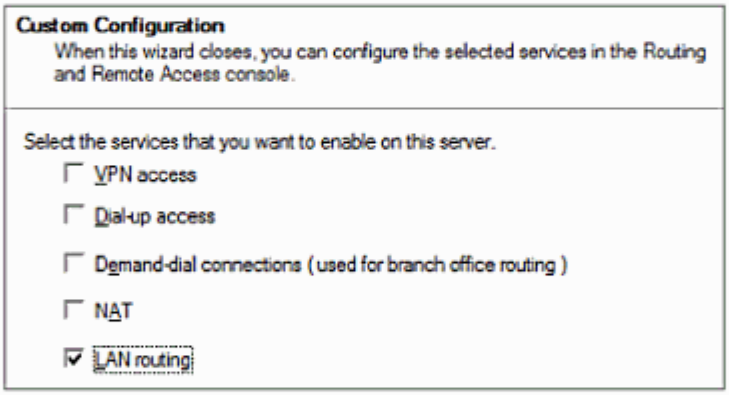
Material cu drept de autor



4-7: Configurarea RRAS

Printre opţiunile diponibile iniţial se enumeră configurarea VPN sau NAT. Pentru configurarea rutării se va alege opţiunea *Custom Configuration* (figura 4-7).

În continuare, din meniul *Custom Configuration* se alege *LAN routing* (figura 4-8).

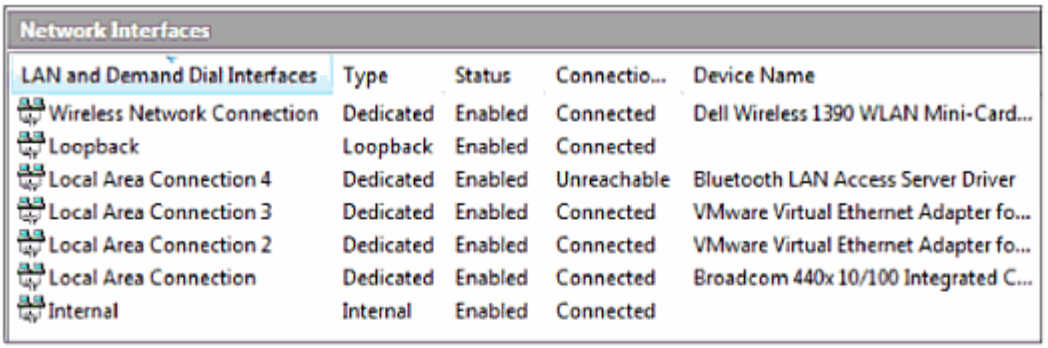


4-8: Activarea rutării

Pentru ca rutarea să se poată realiza, un server trebuie să aibă conectivitate de nivel 3 cu toate reţelele în care are câte o interfaţă deja configurată. De aceea, după conceperea schemei de adrese ce va fi implementată, trebuie configurată fiecare interfaţă de reţea ce va interveni în procesul de rutare cu câte o adresă statică din subreţeaua corespunzătoare ei, împreună cu masca de reţea adecvată. De asemenea, staţiile din reţelele interconectate de către server trebuie să aibă interfaţa acestuia, cea care se află în aceeaşi subreţea cu respectiva staţie, configurată ca default gateway pentru a putea comunica în afara reţelei proprii. Configurarea statică a parametrilor de reţea, atât pentru IPv4 cât şi pentru IPv6 se face din fereastra de proprietăţi a fiecărei conexiuni, după cum s-a explicat în secţiunea 3.4.4.

Lista interfeţelor, a tipurilor conexiunilor şi a stărilor lor pot fi vizualizate din *Server Manager*, în cadrul rolului *Routing and Remote Access*, selectând *Network Interfaces* (figura 4-9).

Material cu drept de autor

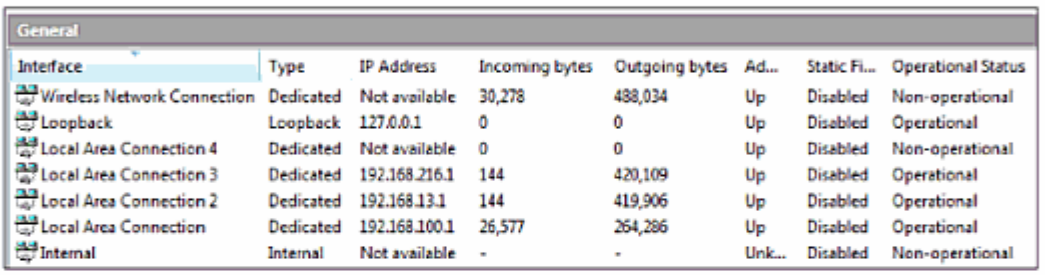


Network Interfaces

LAN and Demand Dial Interfaces	Type	Status	Connectio...	Device Name
Wireless Network Connection	Dedicated	Enabled	Connected	Dell Wireless 1390 WLAN Mini-Card...
Loopback	Loopback	Enabled	Connected	
Local Area Connection 4	Dedicated	Enabled	Unreachable	Bluetooth LAN Access Server Driver
Local Area Connection 3	Dedicated	Enabled	Connected	VMware Virtual Ethernet Adapter fo...
Local Area Connection 2	Dedicated	Enabled	Connected	VMware Virtual Ethernet Adapter fo...
Local Area Connection	Dedicated	Enabled	Connected	Broadcom 440x 10/100 Integrated C...
Internal	Internal	Enabled	Connected	

4-9: Starea interfețelor de rețea din sistem

Pentru a afișa informațiile de nivel 3, împreună cu statistici de trafic pentru fiecare interfață și mai multe detalii despre starea conexiunilor, se poate selecta *General* din cadrul proprietăților de tip IPv4 sau IPv6, sub *Routing and Remote Access* (figura 4-10).



General

Interface	Type	IP Address	Incoming bytes	Outgoing bytes	Ad...	Static Fi...	Operational Status
Wireless Network Connection	Dedicated	Not available	30,278	488,034	Up	Disabled	Non-operational
Loopback	Loopback	127.0.0.1	0	0	Up	Disabled	Operational
Local Area Connection 4	Dedicated	Not available	0	0	Up	Disabled	Non-operational
Local Area Connection 3	Dedicated	192.168.216.1	144	420,109	Up	Disabled	Operational
Local Area Connection 2	Dedicated	192.168.13.1	144	419,906	Up	Disabled	Operational
Local Area Connection	Dedicated	192.168.100.1	26,577	264,286	Up	Disabled	Operational
Internal	Internal	Not available	-	-	Unk...	Disabled	Non-operational

4-10: Starea configurațiilor IPv4 a interfețelor de rețea

4.7.1.1 Configurarea rutelor statice

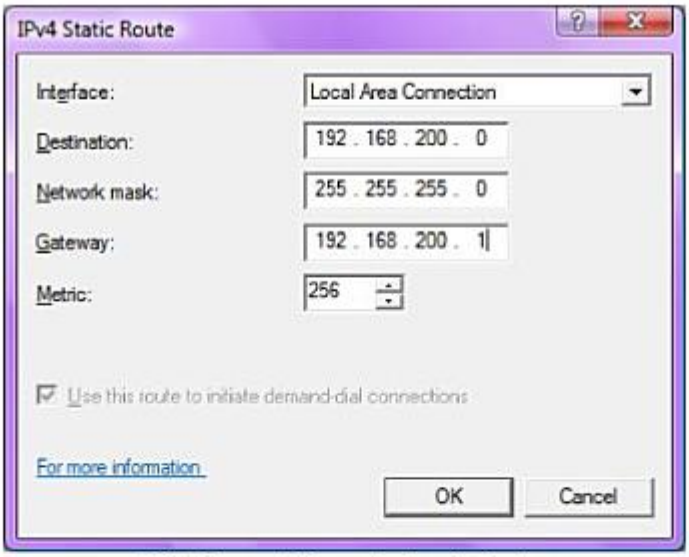
O rută statică reprezintă o rută introdusă manual de către administrator, fixă, ce nu reacționează la schimbările din topologie. În Windows, ea este definită prin:

- o interfață, locală serverului, pe care ruta va fi instalată;
- o adresă destinație, care poate fi adresa unei stații sau adresa unei subrețele;
- o mască de rețea, folosită împreună cu adresa anterioară pentru a defini exact destinațiile rutei;
- un *gateway*, adică adresa dispozitivului care realizează conexiune între rețele. Interfața specificată ca *gateway* este cea care va primi pachetele ce vor utiliza această rută;
- metrică, o metodă de a „măsura" rutele și, în primul rând, pentru a diferenția rute multiple spre aceeași destinație.

Pentru a crea o rută statică se alege întâi categoria, IPv4 sau IPv6 din cadrul rolului *Routing and Remote Access* iar din panoul de acțiune se alege `More Actions > New Static Route` și se completează parametrii descriși mai sus.

După creare, rutele statice vor apărea listate la categoria corespunzătoare (IPV4 sau IPv6), pot fi editate prin dublu-clic și șterse selectându-le și apăsând *Delete* în panoul de acțiuni. De asemenea, rutele devin active imediat ce au fost configurate.

Material cu drept de autor



4-11: Parametrii unei noi rute statice

Static Routes

Destination	Network ma...	Gateway	Interface	Metric	View
192.168.200.0	255.255.255.0	192.168.200.1	Local Area Connection	256	Both
192.168.100.0	255.255.255.0	192.168.100.1	Wireless Network Connection	256	Both

4-12: Listarea rutelor statice

4.7.1.2 Configurarea rutării dinamice

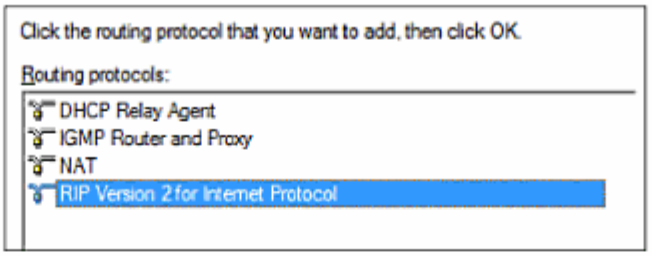
În vederea configurării rutării dinamice, *Windows Server 2008* suportă doar protocolul de rutare RIP (*Routing Internet Protocol*), versiunea 2 (RIP v2). *Windows Server 2003* suporta şi protocolul OSFP (*Open Shortest Path First*) dar acesta a fost eliminat din *Windows Server 2008*, singura opţiune pentru propagarea dinamică a rutelor rămânând utilizarea lui RIP.

Protocolul RIP funcţionează in felul următor : trimite tuturor vecinilor care rulează acelaşi protocol întreaga tabelă de rutare la fiecare 30 de secunde, fiecare client actualizându-şi la rândul său propria tabelă de rutare dacă apar modificări în datele primite de la vecini. Acest mod de funcţionare nu îl face adecvat pentru infrastructuri mari, în care convergenţa rapidă şi eficientă este o cerinţă importantă. Totuşi, nici rutarea în software nu este recomandată în aceste cazuri, deci pentru situaţiile în care *Windows Server 2008* se pretează ca router, este acceptabilă şi performanţa pe care o oferă RIP.

Pentru a adăuga serverului protocolul de rutare RIP se urmează paşii:

1. Se expandează unul dintre nodurile IP din cadrul *Routing and Remote Access*, din *Server Manager*. Se poate alege atât IPv4 cât şi IPv6 pentru a propaga rute folosind RIP.
2. Se face clic dreapta pe *General* şi se alege *New Routing Protocol* din meniu. La fel, opţiunea este disponibilă şi în panoul de acţiuni. Este afişată o listă asemănătoare cu cea din figura 4-13).

Material cu drept de autor



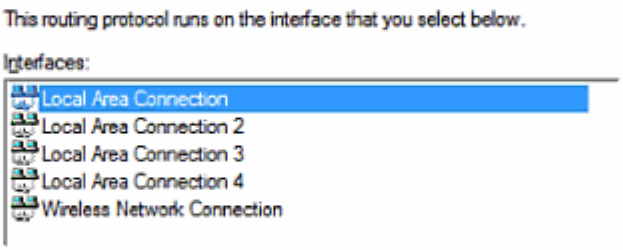
4-13: Selectarea RIP v2 ca protocol de rutare

3. Se selectează RIP din listă după care este adăugat automat un nou nod denumit RIP în cadrul versiunii IP selectate. Selectarea lui RIP pentru IPv4, spre exemplu, nu va configura automat folosirea RIP şi pentru IPv6. Dacă se doreşte folosirea adreselor IPv6, RIP trebuie adăugat explicit pentru acestea.

4.7.1.2.1 Interfețe RIP

Pentru ca RIP să poată funcționa, trebuie să i se comunice pe ce interfețe să ruleze. Mai mult, toate setările ulterioare care influențează comportamentul RIP-ului se fac la nivel de interfață.

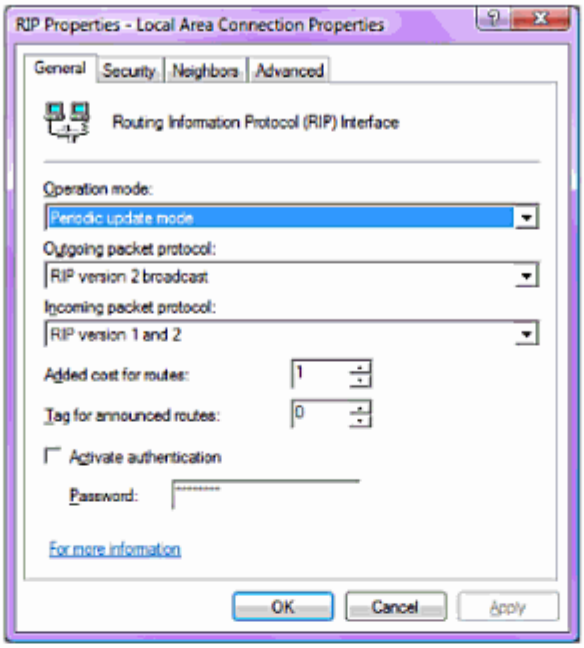
1. Având nodul RIP selectat (creat după procedura descrisă mai sus), se poate face clic dreapta pe acesta, alegându-se opțiunea *New Interface* (sau din panoul de acțiuni) şi se alege o interfață pe care protocolul RIP va rula (figura 4-14).



4-14: Interfaţa pe care va rula RIP

2. La apăsarea butonului OK, este afişată fereastra de proprietăți a interfeței selectate (figura 4-15).
3. Pagina de proprietăți generale oferă posibilitatea selectării versiunii de RIP folosite atât în mesajele acceptate cât şi în cele trimise. Folosirea versiunii 1 implică o serie de limitări suplimentare dar este necesară dacă alte echipamente din rețea o suportă doar pe aceasta.
4. Pagina de securitate permite setarea unor parametri legați de acceptarea rutelor. Există posibilitatea de a accepta toate rutele, sau de a accepta sau ignora rutele dintr-un anumit interval.
5. Setările legate de vecinii cu care se efectuează schimbul de rute includ posibilitatea de a folosi mesaje *multicast* (implicit) sau a celor *unicast* pentru a contacta anumiti vecini (fie în locul mesajelor *multicast*, fie împreună cu ele).
6. Pagina de proprietăți avansate permite modificarea unor parametri ca frecvența de trimitere a actualizărilor, activarea sau dezactivarea unor facilități ca *split-horizon* sau *poison-reverse*, etc.

Material cu drept de autor



4-15: Proprietăţile unei intefeţe RIP

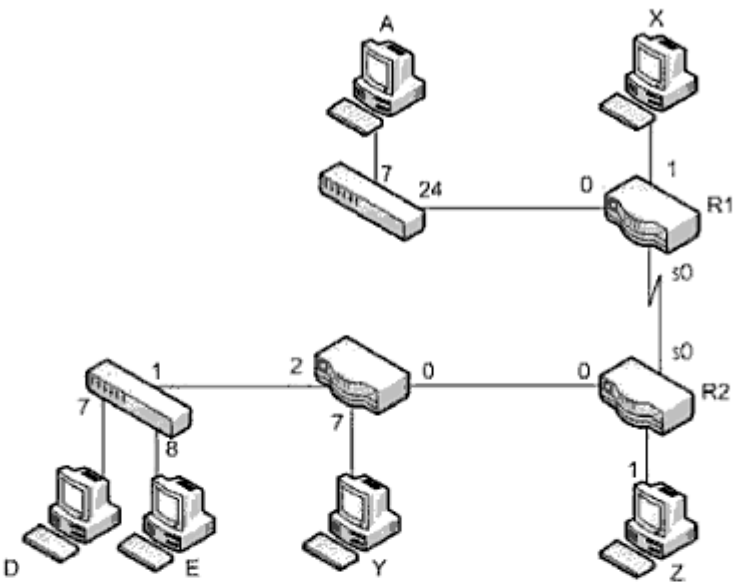
După validarea configurării, interfaţa va apărea în lista intefeţelor active pentru RIP. Pot fi adăugate oricâte alte interfeţe repetând aceeaşi procedură.

Monitorizarea activităţii de rutare se realizează din aceeaşi consolă administrativă. Spre exemplu, pentru a vizualiza o listă a vecinilor care rulează RIP, se poate selecta nodul RIP din una dintre categoriile IPv4 sau IPv6 şi apoi se poate alege opţiunea *Show Neighbors*. Starea interfeţelor incluse în procesul de rutare poate fi vizualizată prin nodul *General* din cadrul protocolului IP corespunzător. De aici poate fi vizualizată tabela de rutare pentru fiecare interfaţă, prin selectarea uneia şi alegerea opţiunii *Show IP Routing Table* din meniul contextual.

Material cu drept de autor

4.8 Studii de caz

4.8.1 Încapsularea pachetelor: exemplificare port forwarding



4-16: Încapsularea pachetelor

În rețeaua dată, pentru cele două rețele cu switchuri s-au folosit adrese private, astfel `R1` și `R3` vor asigura translatare de adresă cu supraîncărcare (PAT). Se va considera că tabelele ARP din rețea au fost configurate **static** pentru toate destinațiile. Descrieți antetele pachetelor apărute în rețea în cazul în care `A` face o singură cerere HTTP către stația `E`.

Rezolvare:

Pentru topologia de mai sus principala problemă o reprezintă asigurarea accesibilității stației E din exteriorul rețelei private.

Dacă ar fi fost disponibil un rezervor de adrese publice pentru a realiza translatarea de adresă, soluția ar fi fost implementarea SNAT (*static NAT*): definirea unei asocieri de translatare pe routerul `R3` între una dintre adresele publice din rezervor și adresa privată a lui `E`. Pentru problema de față, nu dispunem de adrese publice suplimentare, translatarea fiind realizată prin supraîncărcarea adresei publice `<R3, e0>`.

Routerul `R3` asigură translatarea cu supraîncărcare, singura opțiune în acest caz fiind implementarea *port forwarding*, adică crearea pe `R3` a unei asocieri între perechea adresă publică a lui `R3`, port și perechea adresă privată a lui `E` și port. Spre exemplu, dacă se dorește rularea unui serviciu de web pe stația `E`, în serviciul de nume va fi publicată asocierea dintre numele domeniului (fie acesta www.test.com) și adresa publică a lui R3 (adică `IP R3(e0)`). Apoi este definită pe `R3` asocierea de translatare: `<IP R3(e0), 80> - <IP E, 80>`. În acest moment, orice cerere primită de `R3` pe portul `80` nu va mai fi trecută către nivelul aplicație (inclusiv routerul ar putea rula un server de web local), ci va fi translatată și trimisă către stația `E`.

Revenind la problema de față, datorită populării statice a tabelelor ARP pentru toate destinațiile nu vor mai exista pachete de cerere/răspuns ARP. Popularea statică a tabelelor

Material cu drept de autor

ARP este folosită în practică doar ca o metodă de prevenire a unor atacuri [vezi cap 13] [ref], dar datorită scalabilității greoaie este rareori implementată.

Se presupune că în rețea este folosit *default gateway*.

Switchurile vor comuta pachetele fără a le altera, în vreme ce fiecare router va opera cel puțin schimbarea antetului de nivel 2; astfel pachetul de la `A` la `E` va suferi 3 rescrieri.

1. Pachetul trimis de `A` către `E` (adică în afara rețelei locale) va fi destinat la nivel 2 routerului de ieşire, iar la nivel 3 adresei obținute în urma rezolvării de nume, adică adresei publice a lui `R3`. Pachetul fiind unul de cerere HTTP portul destinație va fi 80:

MAC[A]	MAC[R1(e0)]	IP[R3(e0)]	IP[A]	80	9494	date

2. Pachetul ajuns pe `R1` va fi translatat, astfel că pe lângă rescrierea antetului de nivel 2, se vor rescrie câmpurile sursă de nivel 3 şi 4:

FFFF:FFFF:FFFF	IP[R3(e0)]	IP[R1(s0)]	80	43911	date

3. Ruterul `R2` nu va realiza decât rescrierea antetului de nivel 2:

MAC[R3(e0)]	MAC[R2(e0)]	IP[R3(e0)]	IP[R1(s0)]	80	43911	date

4. Ruterul `R3` va realiza operația efectivă de *port forwarding*:

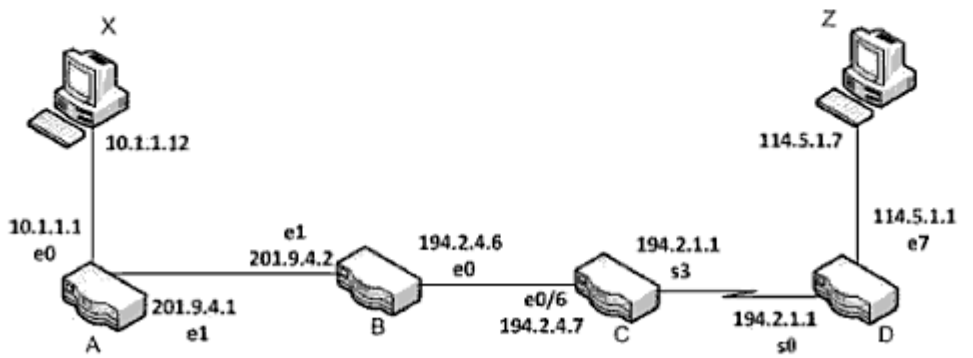
MAC[E]	MAC[R3(e2)]	IP[E)]	IP[R1(s0)]	80	13821	date

Implementarea *port forwarding* nu impune conservarea portului destinație, altfel spus stația `E` poate rula serverul de web pe orice alt port, iar translatarea va reflecta asocierea. Dacă serverul de web ar fi fost configurat să asculte pe portul 999 singura modificare față de analiza de mai sus ar fi definirea translatării pe `R3` (`<IP R3(e0), 80> - <IP E, 999>`) şi antetul pachetului 4:

MAC[E]	MAC[R3(e2)]	IP[E)]	IP[R1(s0)]	999	13821	date

4.8.2 Încapsularea pachetelor: exemplu de tunelare

Fie topologia din figură:



4-17: Exemplu de tunel

Material cu drept de autor

Ruterul A va asigura translatarea de adrese pentru întreaga rețea 10.1.1.0/24, iar routerul B va tunela tot traficul din rețeaua 201.9.4.0/24 și îl va trimite prin interfața sa virtuală tunnel0. Tunelul este stabilit între 194.2.4.6 și 194.2.1.1. În plus, rutarea este asigurată folosind rute statice atfel: pe routerele A și D rutele sunt precizate prin adresa următorului hop, iar pe B și C rutele sunt precizare doar prin interfața de ieșire.

Se consideră că în urma unei pene de curent toate echipamentele sunt proaspăt reinițializate. Stația X va accesa un server de web aflat pe stația Z. Care vor fi antetele tuturor cadrelor ce vor fi trimise în rețea pentru a livra cererea emisă de stația X la stația Z?

Pachetul de cerere web va trece prin 5 rețele și pentru a fi livrat vor generate 13 cadre:

FFFF:FFFF:FFFF	MAC[X]	10.1.1.1	10.1.1.12	Date		
MAC[X]	MAC[A(e0)]	10.1.1.12	10.1.1.1	Date		
MAC[A(e0)]	MAC[X]	114.5.1.7	10.1.1.12	80	55555	Date
FFFF:FFFF:FFFF	MAC[A(e1)]	201.9.4.2	201.9.4.1	Date		
MAC[A(e1)]	MAC[B(e1)]	201.9.4.1	201.9.4.2	Date		
MAC[B(e1)]	MAC[A(e1)]	114.5.1.7	201.9.4.1	80	51311	Date
FFFF:FFFF:FFFF	MAC[B(e0)]	194.2.1.1	194.2.4.6	Date		
MAC[B(e0)]	MAC[C(e0/6)]	194.2.4.6	194.2.1.1	Date		
MAC[C(e0/6)]	MAC[B(e0)]	194.2.1.1	194.2.4.6	114.5.1.7	201.9.4.1	Date
FF		194.2.1.1	194.2.4.6	114.5.1.7	201.9.4.1	Date
FFFF:FFFF:FFFF	MAC[D(e7)]	114.5.1.7	114.5.1.1	Date		
MAC[D(e7)]	MAC[Z]	114.5.1.1	114.5.1.7	Date		
MAC[Z]	MAC[D(e7)]	114.5.1.7	201.9.4.1	Date		

După primirea pachetului de date de la routerul A, routerul B trebuie să îl trimită către ieșirea din tunel, și anume către adresa 194.2.1.1. Routerul B nu știe însă adresa IP a următorului hop, ci doar interfața de ieșire prin care trebuie să trimită pachetul. De aceea, prin interfața Ethernet0 va emite „inocent" o cerere ARP pentru a afla adresa fizică corespunzătoare IP-ului 194.2.1.1. Routerul C, care rulează *proxy ARP* își dă seama (prin verificare cu masca de rețea) că adresa IP destinație pentru care B caută adresa fizică nu se află în aceeași rețea și că cererea ARP nu va ajunge niciodată la 194.2.1.1, și de aceea îi va întoarce lui B un răspuns ARP în care „minte" că adresa MAC corespunzătoare ip-ului 194.2.1.1 este chiar adresa MAC a interfeței sale pe care a primit cererea (și anume e0/6). Pachetul ajuns la C va conține adresele reale destinație și sursă „ascunse" în secțiunea de date. Acest pachet va transmis pe legatura serială păstrând ca adresa IP sursă intrarea în tunel și adresă IP destinație ieșirea din tunel (ca și cum C nu ar exista).

Material cu drept de autor

Întrebări

1. Existența unei intrări în tabela de rutare de forma 141.85.37.0/24 141.85.254.37 înseamnă:

- *trimite toate pachetele venite din rețeaua 141.85.37.0/24 către 141.85.254.37*
- *trimite toate pachetele venite către rețeaua 141.85.37.0/24 către 141.85.254.37*
- *trimite toate pachetele venite din rețeaua 141.85.37.0/24, sau din orice altă subrețea cuprinsă în acest spațiu de adrese, către 141.85.254.37*
- *trimite toate pachetele venite către rețeaua 141.85.37.0/24, sau către orice altă subrețea cuprinsă în acest spațiu de adrese, către 141.85.254.37*

2. Decizia de a ruta sau nu un pachet se bazează pe:

- *Adresa sursă*
- *Adresa destinație*
- *Adresa sursă și portul sursă*
- *Adresa destinație și portul destinație*

3. Fie tabela de rutare de mai jos:

Adresă rețea	Mască	Next hop	Interfață
194.230.85.0	/26	172.17.0.9	E0
200.230.85.128	/26	-	S0
194.230.85.0	/24	194.230.5.65	E1

Și o nouă rută: 199.230.5.0/30 S1. Noua rută va fi inserată pe:

- *prima poziție*
- *după prima rută deja existentă*
- *după cea de a doua rută deja existentă*
- *ultima poziție în tabela de rutare*

4. Către ce interfață va fi rutat un pachet destinat pentru 171.15.68.0 dacă tabela de rutare este cea de mai jos:

Adresă rețea	Mască	Next hop	Interfață
171.15.63.0	/24	172.17.0.9	S0
171.15.64.0	/23	-	S1
0.0.0.0	/0	194.230.5.65	S2

- *va fi trimis pe S0*
- *va fi trimis pe S1*
- *va fi trimis pe S2*
- *nu va fi rutat*

5. Fie două routere. Primul are următoarele adrese asignate: 7.1.1.1/24 – S0 și 7.1.3.1 – S 1, al doilea 7.1.3.2/24 – S0 și 7.1.2.1 – S1. Care este numărul minim de rute ce trebuie adăugat pe routerul 1 pentru a asigura comunicația între 7.1.1.0/24 și 7.1.2.0/24?

- *7.1.2.1 / 24 S1*
- *tot ce e mai sus plus: 7.1.2.1 /24 S0*
- *tot ce e mai sus plus: 0.0.0.0 /0 S1*
- *doar: 0.0.0.0 /0 S1*

Material cu drept de autor

6. Care dintre următoarele descrie cel mai bine convergența rețelei?

- *când mesajele ating simultan un router și apare o coliziune*
- *când mai multe routere rutează simultan pachete pe aceeași cale*
- *când mai multe routere într-o rețea au aceleași cunoștințe despre structura și topologia rețelei*
- *când mai multe mesaje sunt trimise spre aceeași destinație*

7. Timpul cel mai redus de convergență îl au:

- *protocoalele de rutare statică*
- *protocoale bazate pe vectori de distanță*
- *protocoalele bazate pe starea conexiunii*
- *protocoalele de rutare inter-AS*

8. Care este metrica folosită de RIP pentru determinarea căii optime?

- *numărul de routere până la destinație*
- *încărcarea lățimii de bandă*
- *lățimea de bandă până la destinație*
- *niciunul dintre răspunsurile de mai sus*

9. Fie NET1, o rețea direct conectată la un router. Acest router rulează RIP și OSPF și primește de la un router adiacent în cadrul procesului de actualizare informații despre NET1, atât prin RIP, cât și prin OSPF. Câte rute către NET1 vor fi în tabela de rutare la încheierea procesului de actualizare?

- *0*
- *1*
- *2*
- *3*

10. Care este metrica folosită de BGP?

- *numărul de routere până la destinație*
- *încărcarea lățimii de bandă*
- *lățimea de bandă până la destinație*
- *niciunul dintre răspunsurile de mai sus*

Material cu drept de autor

5 Wireless

„The wireless *telegraph is not difficult to understand. The ordinary telegraph is like a very long cat. You pull the tail in New York, and it meows in Los Angeles. The* wireless *is the same, only without the cat."*

Albert Einstein

Ce se învață din acest capitol?

- Funcționarea tehnologiei *wireless* la nivel fizic
- Standardele *wireless* pentru rețele locale
- Implementarea rețelelor *wireless*
- Securitatea în mediul *wireless*
- Configurări pentru rețele *wireless* în Linux
- Configurări pentru rețele *wireless* în Windows

Cine este...

Nikola Tesla *este un inventator și un om de știință în domeniul electricității. Mai mulți biografi contemporani îl consideră pe Tesla „omul care a inventat secolul 20". Patentele și descoperirile teoretice ale lui Tesla au format baza sistemelor de curent electric alternativ. În 1893 a demonstrat pentru prima oară posibilitatea comunicației fără fir (unde radio). Este considerat descoperitorul comunicației fără fir.*

5.1 Introducere în rețele wireless

5.1.1 Introducere în comunicarea *wireless*

Pe fondul unei nevoi de mobilitate şi conectivitate din ce în ce mai crescute, comunicația fără fir a înregistrat o explozie fulminantă în ultimii ani. Răspândirea dispozitivelor mobile (calculatoare *notebook*, PDA-uri sau *smartphone*-uri) este cea care a condus în mare măsură la dezvoltarea tehnologiilor de comunicație fără fir, fără a fi însă singurul motor. Tendința de migrare spre digital pentru o gamă din ce în ce mai largă de dispozitive generează de asemenea o nevoie de interconectare crescută. Pentru acestea, perspectiva comunicației fără fir este foarte atrăgătoare. Motivele principale sunt mobilitatea crescută şi reducerea costurilor pentru dezvoltarea infrastructurii. Deşi în trecut securizarea unei rețele *wireless* se dovedise a fi o provocare la care organizațiile de standardizare încă nu răspunseseră, în prezent există mai multe protocoale standardizate care pot oferi o securitate sporită.

Cu toate acestea, tehnologia *wireless* nu are rolul să o înlocuiască pe cea cu fir, ci mai degrabă să o completeze. Motivul principal este lățimea de bandă: în timp ce *Ethernet*-ul a ajuns să ofere 10 *Gbps*, tehnologiile *wireless* actuale nu depăşesc 54 Mbps (600 Mbps în cazul tehnologiilor în curs de standardizare).

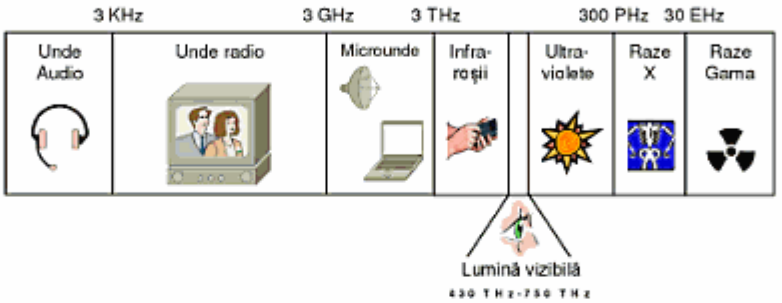
În plus, datorită utilizării switchurilor, tehnologia *Ethernet* asigură o comunicație full-duplex. Cu alte cuvinte, dacă zece stații cu plăci *Ethernet* de 10 Mbps sunt conectate în acelaşi switch şi switchul are o capacitate de comutare destul de mare, fiecare stație va putea avea garantată, banda de 10Mbps în rețeaua locală. De partea cealaltă, standardul *wireless* permite unei singure stații să transmită la un moment dat. Aceasta este o limitare a mediului fizic şi a proiectării tehnologiei, deoarece, în rețele *wireless*, un dispozitiv foloseşte aceeaşi frecvența a semnalului si pentru transmisie si pentru recepție. Este important de reținut faptul că în rețele *wireless*, banda maximă se împarte la numărul de stații.

Material cu drept de autor

5.1.2 Considerente de nivel fizic

5.1.2.1 Mediul fizic

Baza fizică pentru comunicația fără fir o reprezintă undele electromagnetice, folosite în frecvențele cele mai potrivite pentru transmisii de date. Mediul fizic de propagare al undelor electromagnetice nu este de fapt necesar, căci un semnal *wireless* se poate propaga fără probleme și în vid (de aceea căldura și lumina soarelui ajung pe Pământ). Când vorbim despre comunicația de date, principala proprietate a undelor este frecvență. Funcție de frecvența pe care o tehnologie funcționează, se poate determina calitatea semnalului în atmosferă, interferența cu alte dispozitive, distanța de propagare a semnalului și chiar și lățimea de bandă. În figura de mai jos este reprezentat spectrul electromagnetic din punctul de vedere al utilizării undelor de diverse frecvențe în comunicații:



5-1: Benzi de frecvență

Frecvențele folosite în rețele de calculatoare pentru transmisiile *wireless* se află în intervalele de 2.4 - 2.4835 GHz, 5,725 - 5,850 GHz și recent adăugatul interval de 5.47 - 5.725 GHz. Deci undele electromagnetice prin care se propaga semnalul *wireless* în comunicații de date, sunt **unde radio** de înaltă frecvență și **microunde**.

În folosirea undelor electromagnetice pentru transmisii de date, trebuie avute în vedere următoarele considerente fizice:

- Undele electromagnetice nu sunt clar delimitate din punct de vedere al propagării, precum este semnalul de pe un cablu de cupru protejat de un izolator. Orice dispozitiv care ascultă mediul și este în raza de propagare, poate recepționa semnalul *wireless*. Acesta este un considerent de securitate foarte important.
- Undele electromagnetice nu sunt protejate de semnale exterioare, precum este semnalul de pe un cablu de cupru protejat de un izolator. Odată cunoscută frecvența de transmisie a unei rețele *wireless*, administratorul de rețea trebuie să fie conștient de alte dispozitive sau tehnologii ce funcționează în aceeași frecvență.
- Semnalul *wireless* se atenuează odată cu propagarea prin mediul fizic. Atenuarea poate interveni din mai multe cauze:
  - absorbirea semnalului în atmosferă
  - efecte termodinamice cauzate de căldură sau umiditate sporită
  - interferențe cu alte semnale
  - reflexia parțială a semnalului pe suprafețele diferitelor materiale de construcție
- Din considerente legale, nu se poate transmite în spectru pe o anumită frecvență, fără a avea licență pe acea frecvență. O rețea *wireless*, ar fi greu de implementat atât din punct de vedere birocratic, cât și din punct de vedere al propagării în popularitate a tehnologiei, dacă fiecare utilizator, atunci când și-ar cumpăra un router *wireless*, ar trebui să plătească și pentru licențierea unei benzi de frecvență în care să transmită. În afară de acest aspect, frecvența

Material cu drept de autor

dispozitivelor *wireless* trebuie să fie aceeași în toată lumea rețelelor de calculatoare, pentru ca tehnologia să fie compatibilă.

5.1.2.2 Benzile ISM și UNII

Pentru a efectua o transmisie *wireless* este nevoie de **licență** din partea unei autorități. Spre exemplu, statul român a acordat licență postului de radio Europa FM să transmită în banda de 106,7 MHz pe teritoriul orașului București. Orange și Vodafone au licență pentru a efectua transmisii în benzile: 890 MHz – 960 MHz respectiv 1710 MHz – 1880 MHz.

Exista totuși o metodă de a efectua transmisii fără licență: transmisia în una dintre benzile **ISM** (*Industrial, Scientific and Medical*) sau **UNII** (*Unlicensed National Information Infrastructure*). S-a căzut de acord că la nivel internațional aceste benzi să nu fie licențiate, astfel încât să poată fi utilizate de oricine. Drept consecința, nu este nevoie de licență pentru a acționa telecomanda de la alarmă, pentru a utiliza telefonul *cordless*, sau pentru a folosi un *mouse* sau o tastatură *wireless*. Toate acestea operează în una dintre benzile ISM sau UNII.

Pentru a elimina necesitatea obținerii unei licențe pentru utilizarea rețelelor *wireless*, și acestea funcționează în aceste benzi. Există două benzi ISM de interes pentru tehnologia *wireless*: 2,4 GHz – 2,4835 GHz și 5,725 GHz – 5,850 GHz. Din 2004, odată cu standardul 802.11h, s-a adăugat și banda UNII de 5.47 - 5.725 GHz. Aceasta nouă bandă va fi discutată mai târziu în capitolul de comunicare *wireless* [5.1.6.3]

Notă: *La începuturile tehnologiei wireless, se folosea și frecvența de 900 MHz. Însă, din cauza faptului că doar câteva țări utilizau ofereau posibilitatea de utilizare fără licență, banda a fost retrasă.*

Faptul că se lucrează în benzile ISM aduce însă după sine o constrângere: în aceste benzi este limitată legal posibilitatea de a transmite un semnal de putere mare. Scopul acestei măsuri este de a limita distanța maximă de transmisie în vederea reducerii interferenței între echipamentele diverșilor utilizatori. Astfel, raza de funcționare a rețelelor *wireless* este limitată prin lege.

5.1.2.3 Frecvența 2.4 GHz *vs* 5 GHz

De la apariția primelor tehnologii *wireless*, soluțiile existente se împărțeau între banda de 2.4 GHz și banda de 5 GHz. Fiecare dintre acestea oferă avantaje și dezavantaje de implementare și funcționalitate. Diferențele dintre benzi se reduc de fapt la diferențele între transmisiile în frecvență înaltă și cele în frecvența joasă. În cele ce urmează, se va realiza o comparație între cele două și se va sfârși prin a desemna banda cea mai utilizată în prezent și motivele din spatele adoptării acesteia.

- Distanța de propagare

Undele de frecvența joasă sunt absorbite foarte puțin în atmosferă, de aceea ele pot străbate distanțe foarte mari. Cu cât crește frecvența unei unde, cu atât aceasta este absorbită mai mult în atmosferă. De asemenea, undele joase prezintă o capacitate de penetrare a materialelor foarte mare, fiind cu atât mai potrivite pentru transmisiile de date. Spre deosebire de acestea, undele de frecvențe mari tind să sufere reflexii și refracții pe diverse suprafețe.

Concluzionând, dacă se folosește banda de 2.4 GHz, absorbția în atmosferă o să fie mai mică, reflexia semnalului de asemenea, deci distanța posibilă de propagare o să fie mai mare.

- Lățimea de bandă

Material cu drept de autor

Folosind o frecvenţa mai mare obţinem o creştere aproximativ liniară a lăţimii de bandă. Spre exemplu, folosirea benzii de 900 MHz oferea o bandă de 860 Kbps. Odată cu trecerea la 2.4, banda teoretică a ajuns la valoarea de 2 Mbps.

- Interferenţe cu alte tehnologii

Din nefericire, banda de 2,4 GHz a ajuns să fie destul de aglomerată. În această bandă operează *Bluetooth*, perifericele *wireless*, telecomenzile şi alte dispozitive cu care vor apărea inevitabil interferenţe. Banda de 5 GHz este destul de puţin ocupată însă dezavantajul este absorbţia mai mare a semnalului în mediul fizic.

- Costul echipamentelor

Costul echipamentelor ce funcţionează in banda de 5 GHz sunt sensibil mai scumpe decât cele din banda de 2.4 GHz. Acesta a fost şi unul din motivele pentru care tehnologia de 2.4 GHz a câştigat teren pe piaţa *wireless*.

Tabelul de mai jos sintetizează proprietăţile undelor, prezentate până acum:

Criteriu/Frecvenţă	Frecvenţe mici	Frecvenţe mari
Distanţă	mare	mică
Lăţime de bandă	mică	mare
Interferenţe	mari	mici
Cost	mic	mare

5-2: Proprietăţile undelor electromagnetice

Frecvenţa cel mai des utilizată în prezent în reţelele de date este cea de 2.4 GHz. Deşi mult mai aglomerată, aceasta s-a bucurat de mult suport din parte *third-parties* precum Centrino sau *WI-FI Aliance*. Componenta de business a monopolului benzii de 2.4 GHz va fi discutată în următoare secţiune.

5.1.3 Standarde pentru reţele locale (WLANs)

5.1.3.1 Standardul 802.11b

Lansat în 1999, 802.11b este al doilea protocol ca şi popularitate în zilele noastre. Operează în banda ISM de 2,4 GHz şi atinge o bandă de 11 Mbps. 802.11b este performerul la capitolul **rază** a reţelei: pentru că operează la o frecvenţă mică, aceasta poate ajunge la câteva sute de metri şi penetra până la 4-5 pereţi de beton folosind echipamente convenţionale.

La capitolul interoperabilitate cu alte standarde, un echipament 802.11b poate comunica cu unul 802.11g, dar nu cu unul 802.11a. Deşi acest standard a apărut odată cu 802.11a, a reuşit să se impună prin preţurile echipamentelor mai scăzute şi prin suportul pe care l-a primit din partea *Wi-Fi Alliance*[1]. La apariţia 802.11b, organizaţia a realizat un nou brand pe care l-a popularizat şi l-a promovat, oferind standardului numele de *Wi-Fi*. De asemenea au fost create şi *stickere* speciale care indicau dacă un dispozitiv este sau nu compatibil cu acest standard.

Dezavantajul major ţine, însă, de bandă: 11 Mbps este destul de puţin şi, revenind la discuţia cu banda reală ce se împarte între staţii, se poate spune că 802.11b nu se pretează la reţele mai mari.

[1] Wi-Fi Alliance este un consorţiu format din peste 300 dintre cele mai mari companii din domeniul IT, având ca scop promovarea şi dezvoltarea tehnologiilor *wireless*.

Material cu drept de autor

Se mai foloseşte 802.11b? În general, astăzi, nu prea mai există pe piaţă echipamente exclusiv 802.11b ci doar 802.11b/g. Mai poate fi însă folosit de către dispozitive care au nevoie de acoperire mare şi nu de lăţime de bandă. O serie de sisteme *embedded* intră în această ultimă categorie.

5.1.3.2 Standardul 802.11a

Deşi a apărut ca standard în acelaşi timp cu 802.11b, echipamentele 802.11a nu au apărut pe piaţă decât spre sfârşitul anului 2000. 802.11a este singurul din familia de protocoale WLAN, care operează în frecvenţa de 5 GHz, oferind o lungime de bandă de 54 Mbps. Pentru a putea oferi o viteză mai bună decât *Wi-Fi* la o frecvenţă mai mare, standardul foloseşte o modulare avansată a undei purtătoare de semnal, numită OFDM (*Orthogonal Frequency Division Multiplexing*). Prin intermediul acesteia, microundele se compun într-un mod nedestructiv în aer ocupând mult mai eficient banda oferită de canalul fizic într-un interval măsurat de timp. Crescând cantitatea de informaţie utilă pe secundă, creşte şi lăţimea de bandă pe care tehnologia o pune la dispoziţie.

Avantajele standardului 802.11a sunt, în primul rând, lăţimea de banda şi lipsa interferenţelor.

Părţile mai puţin bune ale tehnologiei sunt tocmai urmarea faptului că se operează la frecvenţe foarte mari. La 5 GHz se află microundele ce au o lungime de undă de doar 6 cm. Astfel de unde se absorb foarte uşor în aer şi se reflectă pe diverse suprafeţe, neputând penetra uşor materialele. O consecinţa interesantă a acestei proprietăţi fizice, este folosirea acestui standard în medii în care se doreşte o securitate sporită la nivel fizic. În general este destul de greu să se asigure controlul propagării undelor electromagnetice, dar dacă reţeaua trebuie amplasată într-un amfiteatru în care suprafeţele pereţilor au un grad de reflexie mare, se preferă standardul 802.11a pentru o mai bună izolare a semnalului la nivelul încăperii.

5.1.3.3 Standardul 802.11g

Până la apariţia 802.11g, foarte multă lume folosea 802.11b. Lansat în 2002, 802.11g combină avantajele 802.11a (banda de 54 Mbps) cu avantajele 802.11b (rază mare de acoperire). Astfel, 802.11g operează în banda ISM de 2,4 *Ghz* şi foloseşte OFDM pentru atingerea ratei de transfer de 54 Mbps.

Succesul 802.11g s-a datorat şi păstrării compatibilităţii cu 802.11b, tehnologia cea mai răspândită până atunci. Toate echipamentele 802.11g sunt compatibile cu cele b. De fapt, acestea pot opera în ambele standarde. Astfel, s-a putut realiza o trecere comodă de la b la g prin înlocuirea treptată a echipamentelor, fără a fi necesară o investiţie mare într-un timp scurt.

Spre deosebire de 802.11a, banda de 54 Mbps nu are un grad atât de mare de constanţă datorită interferenţelor mai mari din banda ISM de 2,4 GHz în raport cu cea de 5 GHz. Însă frecvenţa mai mică permite păstrarea calităţii semnalului pe distanţe mult mai mari, degradarea benzii de 54 Mbps fiind mult mai lentă decât la 802.11a.

Astăzi, 802.11g este câştigătorul absolut în lupta între standarde.

5.1.3.4 Standardul 802.11n

Una din principalele probleme cu care se confruntă reţelele *wireless* actuale este lăţimea de bandă. Cea mai mare lăţime de bandă oferită de un standard WLAN este de 54 Mbps. Astfel, IEEE a creat în 2003 un grup de lucru pentru dezvoltarea unui proiect care să rezolve nevoile tot mai mari de viteză şi stabilitate ale utilizatorilor. Aflat la versiunea 4.0 a *draft*-ului

Material cu drept de autor

proiectului, se aşteaptă ca noul standard să fie finalizat în decursul anului 2008, cel târziu 2009.

Noul standard îşi propune să ofere o bandă de 600 Mbps şi o rază de acoperire de 2-4 ori mai mare decât a standardelor actuale. În acelaşi timp, prin mărirea ratei de transfer şi micşorarea timpului de funcţionare a dispozitivului, se va diminua consumul de energie. Pentru a creşte performanţele, standardul se bazează pe tehnologia MIMO (*Multiple Input Multiple Output*) care foloseşte un sistem de mai multe antene pentru transmisia şi recepţia datelor. O cerinţă importantă pentru noul standard este păstrarea compatibilităţii cu standardele deja existente 802.11a/b/g. Astfel, un echipament 802.11n va putea funcţiona fie în banda de 2,4 GHz, fie în banda de 5 GHz.

În ciuda faptului că standardul nu este deocamdată finalizat, o serie de producători au fost atraşi de performanţele teoretice oferite de noua tehnologie şi au început deja să ofere produse bazate pe versiunea 4.0 ale specificaţiilor temporare.

5.1.4 Wireless MAN

În reţele locale, standardul *wireless* a fost foarte bine primit de către piaţă, cunoscând o evoluţie constantă de-a lungul timpului. Din păcate, lucrurile nu stau deloc astfel în WAN/MAN. În general se presupune ca nu există implementări de *wireless* MAN, sau că acestea există, dar în număr foarte mic. De fapt există multe soluţii *wireless* MAN implementate, însă problema este că 95% din acestea sunt proprietare. Funcţionează cu protocoale dezvoltate intern de către firmele ce deţin reţeaua şi fiindcă fiecare companie îşi dezvoltă propriul set de reguli, bineînţeles că acestea nu pot să funcţioneze (comunice) între ele. Singurul standard *wireless* dezvoltat pentru MAN este 802.16e, sau **WiMAX** (*Worldwide Interoperability for Microwave Access*).

O conexiune Wireless MAN oferă o lăţime de bandă de până la 70 Mbps în cazul *WiMAX*. Aria de acoperire a unei staţii de bază este de până la 50 km în mod teoretic. În mediul urban însă, este posibilă o acoperire de 2-5 km pentru *WiMAX* Mobil. Trebuie să menţionat faptul că vederea directă (LoS – *Line of Sight*) între staţia de bază şi terminal nu este necesară.

Deşi există posibilitatea utilizării unor benzi nelicenţiate de genul celei de 5GHz, în general se folosesc frecvenţe licenţiate pentru a putea avea control asupra spectrului. Pot fi folosite orice frecvenţe din intervalul 2-11 GHz (2-6 GHz în cazul *WiMAX* Mobil), însă instituţiile de standardizare recomandă folosirea benzilor de 2,3 GHz, 2,5 GHz sau 3,5 GHz pentru a putea avea o interoperabilitate a echipamentelor şi, de aici, o scădere a preţurilor.

Un avantaj important al *WiMAX* este faptul că oferă suport pentru calitatea serviciilor (QoS - *Quality of Service*).

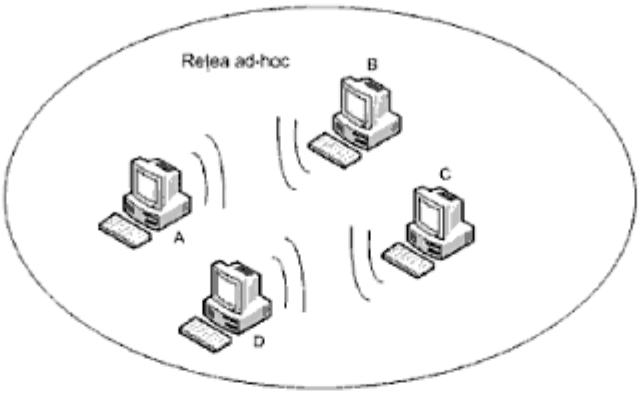
5.1.5 Implementarea reţelelor *wireless*

5.1.5.1 Topologii *wireless*

WLAN-urile se clasifică în două tipuri din punct de vedere al topologiei:

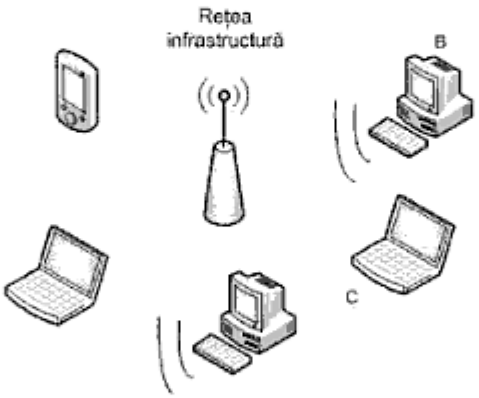
- Reţele *ad hoc*
- Reţele de tip infrastructură

Material cu drept de autor



5-3: Rețea *ad hoc*

O rețea ad hoc este echivalentul în Ethernet *al unei rețele full-mesh, în care fiecare stație este conectată prin interfața* wireless *direct la celelalte stații. Cu alte cuvinte, traficul generat de o stație A, destinat unei stații B, trece direct de la A la B, fără un dispozitiv intermediar.*



5-4: Rețea infrastructură

O rețea de tip infrastructură presupune existența unui dispozitiv central care se ocupă de managementul rețelei wireless și prin care trec toate pachetele din rețea în drumul lor de la sursă, spre destinație. Acest dispozitiv central poate fi un acces point sau un router wireless.

5.1.5.2 Echipamente *wireless* de interconectare

***Access point**-urile, sau mai pe scurt AP-urile, joacă rolul de punct central de comunicație într-o rețea wireless.*

De asemenea, tot ele sunt cele ce interconectează rețelele fără fir cu infrastructura *wired*. Ele dispun de o interfață *wired* pentru interconectarea la rețeaua cu fir și una *wireless* pentru comunicația cu stațiile echipate cu interfețe *wireless*. AP-urile pot fi văzute ca și huburile din rețeaua *Ethernet* din punct de vedere al funcționalității în rețea.

Un AP are două funcții de nivel 2 importante:

- **Asocierea clienților** presupune includerea clienților la nivel 2 în rețeaua *wireless*.

Material cu drept de autor

- **Autentificarea clienţilor** presupune verificarea identităţii clienţilor ce doresc să se conecteze la reţea (în cazul în care se foloseşte un mecanism de securitate specific)

La prima vedere, conceptul de asociere de nivel 2 la o reţea poate suna confuz. Într-o reţea unde protocolul de nivel 2 este *Ethernet*, nu există conceptul de reţea de nivel 2. Într-o topologie 802.3, noţiunea de reţea există doar la nivel superior, la nivel IP. Însă, într-o reţea *wireless*, conceptul de reţea există atât la nivelul protocolului 802.11, cât şi la nivelul protocolului de nivel 3.

O reţea *wireless*, este identificată la nivel 2, de un nume special, denumit în standard: **SSID** *(Service Set Identifier)*. Dacă un client doreşte să se asocieze cu o reţea *wireless* (termenul de asocierea se referă implicit la conectivitate de nivel 2), trebuie să cunoască SSID-ul acestei reţele. Majoritatea clienţilor *wireless* permit scanarea mediului pentru a găsi SSID-ul reţelelor la care se pot asocia. După ce o staţie s-a asociat unei reţele, se va face o **cerere DHCP** pentru a obţine o adresă IP. Răspunsul la această cerere va trebui sa vină fie de la un server DHCP dedicat aflat pe o staţie în reţea, fie direct de la AP (în cazul în care AP-ul are deja integrat un server DHCP).

Routerele *wireless* sunt dispozitive care pot realiza în acelaşi timp funcţiile unui AP, unui switch de nivel 2 şi unui router. Aceste echipamente sunt prevăzute cu:

- O antenă *wireless* - pentru îndeplinirea funcţiilor de AP
- Unul sau mai multe porturi de LAN - aceste porturi sunt de fapt conectate într-un switch care se află înăuntrul routerului *wireless*, acesta fiind transparent pentru utilizatori. Porturile acestea sunt prezente pentru a putea oferi şi funcţionalitate de switch, într-o reţea în care nu toate staţiile au interfeţe *wireless*. Între oricare dintre aceste porturi se face *switching*.
- Un port de WAN – acest port este cel la care se leagă conexiunea de la ISP. Între acest port şi oricare dintre porturile de LAN, se face rutare.

Bridge-urile seamănă foarte bine cu *access point*-urile din punctul de vedere al arhitecturii hardware. Ele sunt folosite însă pentru interconectarea reţelelor printr-o legătură *wireless*. Un exemplu tipic de utilizare a *bridge*-urilor este realizarea unei conexiuni între două clădiri.

Un *bridge*, asemeni unui AP, are o interfaţă *wired* şi una *wireless*: cadrele ce vin pe una dintre interfeţe sunt transmise către cealaltă, cu eventuala translaţie între formatele cadrelor celor două reţele. De asemenea, un *bridge* poate decide să nu transmită mai departe un pachet în cazul în care ştie că destinatarul se află în reţeaua din care a venit pachetul. Un *bridge* însă, nu permite asocierea nodurilor, de aceea, pentru a conecta mai multe staţii dotate cu interfeţe *wireless*, este în continuare nevoie de un *access point*.

5.1.5.3 PoE (Power over *Ethernet*)

În implementarea unei reţele *wireless*, amplasarea AP-ului este foarte importantă. Locul în care se montează dispozitivul central al reţelei nu ar trebui să depindă de nimic altceva în afară de acoperirea cât mai bună a locaţiei. Însă AP-ul nu este un dispozitiv pasiv, ci are nevoie de alimentare de la reţeaua electrică pentru a funcţiona. Din acest motiv, de multe ori AP-ul se instalează de fapt acolo unde există alimentare şi nu în locul din care ar oferi acoperire optimă.

Pentru a rezolva această problemă, multe din switchurile din prezent oferă PoE *(Power over Ethernet)*. Această tehnologie presupune transmiterea de curent electric pe aceleaşi fire pe care se face şi transmisia de date. Ideea nu este una nouă aceasta existând implementată şi în telefonie.

Deşi tensiunea oferită este doar de 48 V, aceasta este îndeajuns pentru dispozitive precum AP-urile sau telefoanele IP. În concluzie, folosind switchuri cu PoE, este de ajuns să se conecteze AP-urile la acestea cu un cablu UTP, şi acestea vor fi în acelaşi timp alimentate şi conectate la reţea.

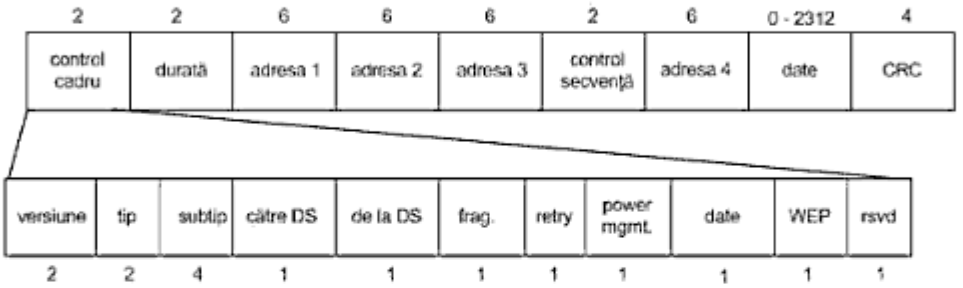
Material cu drept de autor

Dacă se dorește și mai mult minimizarea dependenței față de rețeaua electrică, se poate conecta un UPS[1] la switchul cu PoE, păstrând astfel conectivitatea *wireless* în rețeaua locală și în momentul în care nu există curent electric.

5.1.6 Comunicarea *wireless*

5.1.6.1 Formatul cadrului 802.11

Având în vedere că rețelele *wireless* sunt cel mai adesea continuate prin rețele *Ethernet*, unul din miturile false ale lumii rețelelor de calculatoare este ca formatul cadrului este identic în ambele standarde. În realitate modul de funcționare și cerințele unei rețele *wireless*, sunt destul de diferite de ceea ce presupune standardul *Ethernet*. În continuare se vor expune asemănările dintre cele 2 formate, punându-se accent pe cadrul 802.11.



5-5: Formatul cadrului 802.11

> ***Notă:*** *în desenul de mai sus, cifra ce se află deasupra fiecărui câmp din cadru specifică numărul de octeți ocupat în antet iar mărimea câmpurilor de sub „control cadru" este exprimată în biți. Specificațiile din figura au fost extrase din standardul republicat de IEEE în 2007.*

Prima observație este că dimensiunea maximă posibilă a cadrului *wireless* este de 2346 de octeți. După cum s-a specificat în capitolul 2, protocolul *Ethernet* nu permite un MTU mai mare de 1518 octeți. Ce se va întâmpla deci, când un cadru *wireless* de dimensiunea mai mai mare de 1518 *bytes*, va intra într-o rețea *Ethernet*? Cum la nivelul 2, protocolul *Ethernet* nu oferă o posibilitate de fragmentare a unui cadru de dimensiune prea mare, protocolul de nivel superior (cel mai adesea este vorba de IP) va trebui să ofere serviciul de fragmentare. Bineînțeles ca acest proces va avea întotdeauna loc în interiorul unui *bridge* sau AP.

> ***Notă:*** *în general MTU-ul de pe rețele wireless este setat implicit la maxim 1518 pentru a evita procesul de fragmentare care introduce un overhead de procesare la nivelul echipamentelor de rețea.*

Primul câmp din cadrul *wireless* este numit cadrul de control și ocupă 2 octeți. Cele mai importante informații pe care acesta le furnizează sunt biții de *To DS*[2] și *From DS*. Cele 4 combinații care se pot obține din varierea valorilor acestor 2 biți oferă o interpretare unică a celor 4 adrese MAC din cadrul *wireless*. În continuare se vor prezenta aceste interpretări:

[1] *Uninterruptible power supply* – dispozitive care în cazul unei pane de curent, pot oferi energie electrică, fără a întrerupe fluxul de alimentare.

[2] *Distribution System* – termenul este echivalent cu access point.

Material cu drept de autor

către AP	de la AP	adresa 1	adresa 2	adresa 3	adresa 4
0	0	destinație	sursă	BSSID	-
0	1	destinație	BSSID	sursă	-
1	0	BSSID	sursă	destinație	-
1	1	receptor	transmițător	destinație	sursă

5-6: Utilizarea câmpurilor "către AP" şi "de la AP"

Pentru o mai bună înțelegere a câmpurilor de mai sus, se recomandă consultarea standardului IEEE publicat în 2007[1].

In interiorul cadrului 802.11 este prezent şi un câmp de durată. Acesta fusese creat pentru a oferi unei stații posibilitatea de a putea comunica celorlalte stații perioada în care aceasta va ocupa mediul. Se va analiza pe scurt un scenariu simplu pentru a releva importanța acestei facilități.

1. Se presupune că într-o rețea *wireless*, stația A vrea să transmită.
2. Aceasta va asculta mediul (*Carrier Sense*) şi dacă este liber (*Multiple Access*) va începe să transmită completând şi câmpul de durată cu o valoare estimată.
3. Cadrul pe care stația A îl va transmite va ajunge la toate stațiile care sunt în raza sa de transmisie.
4. Stația B asculta şi ea mediul, căci vroia să transmită. Când va primi cadrul stației A, va citi în câmpul de durată că aceasta va ocupa mediul timp de x secunde.
5. Ştiind că timp de x secunde mediul va fi ocupat cu transmisia stației A, stația B nu va mai scana mediul în acest interval şi astfel va conserva, astfel, putere.

La începuturile standardului, ideea era una destul de practică, căci dispozitivele care sunt în general dotate cu capabilități *wireless*, au acumulatori cu timp limitat de funcționare. S-a observat însă, că informația de durata poate deschide rețeaua la unele hibe de securitate şi de aceea câmpul de durată nu este folosit în comunicațiile din prezent.

5.1.6.2 Accesul la mediu

Rețelele 802.11 se mai numesc, eronat, „*Wireless Ethernet*". Deşi cele două tehnologii au unele lucruri în comun, asemănări majore între ele nu există nici la nivel fizic, nici la nivelul legătură de date.

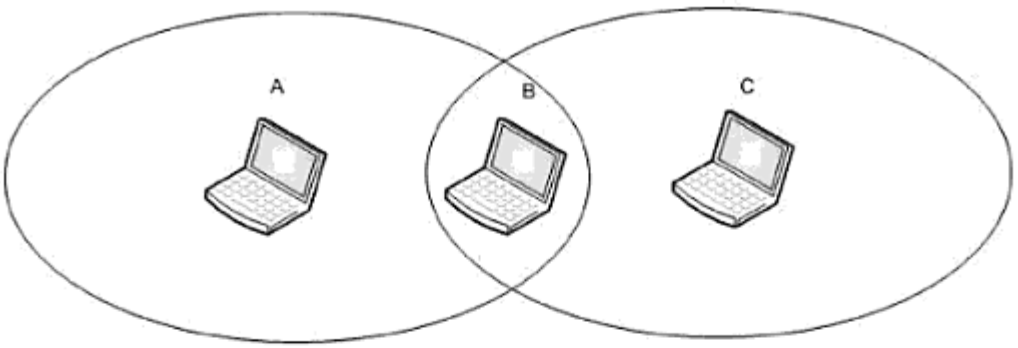
În specificația *Ethernet* tehnica de acces la mediu este CSMA/CD. Aceasta însă se referă la situația în care mediul este comun, adică fie se utilizează huburi, fie topologiile erau de tip magistrală. Astăzi însă, niciuna dintre situațiile acestea nu mai este întâlnită în practică: se utilizează switchuri ce elimină coliziunile de pachete pe mediu, aceasta făcând CSMA/CD inutilă. Printre beneficii se numără atât comunicația *full-duplex* cât şi banda garantată.

În cazul rețelelor *wireless* ne întoarcem însă la situația în care mediul este partajat, din acest punct de vedere, rețeaua fără fir semănând cu o rețea *Ethernet* bazată pe hub. Anumite particularități fac însă CSMA/CD să fie ineficientă în cazul transmisiilor fără fir.

Să considerăm următorul exemplu: trei stații A, B şi C fac parte dintr-o rețea *wireless adhoc*. B este în raza lui A dar C nu. A dorește să transmită către B.

[1] http://standards.ieee.org/getieee802/download/802.11-2007.pdf

Material cu drept de autor



5-7: Problema staţiei ascunse

Dacă se foloseşte CSMA/CD, atunci A va asculta mediul şi dacă nu recepţionează semnal, va începe să transmită. Ce se întâmplă dacă C transmite către B în momentul acesta? A nu va recepţiona semnalul pentru că C nu e în raza sa de acoperire, aşa că va începe să transmită odată cu staţia C, rezultând astfel o coliziune pe mediu. Această problemă se numeşte **problema staţiei ascunse**.

Pe lângă acest gen de probleme, mai apare încă una, de ordin tehnic: un transceiver radio nu poate transmite şi recepţiona în acelaşi timp. Aşa că o staţie **nu poate asculta mediul în timp ce transmite**, pentru a vedea dacă a apărut vreo coliziune.

5.1.6.2.1 CSMA/CA

În concluzie, nu putem folosi CSMA/CD pentru a asigura accesul la mediul *wireless* partajat. Soluţia adoptată de standard se numeşte CSMA/CA *(Carrier Sense Multiple Access/Collision Avoidance)*. Funcţionarea acestui mecanism se bazează pe trimiterea unui cadru special de *ACK* de la destinaţie la sursă, după fiecare cadru 802.11 primit la destinaţie. Dacă după trimiterea unui cadru, nu se primeşte un *ACK*, se aşteaptă un timp aleator şi se încearcă din nou să se trimită cadrul pentru care nu s-a primit *ACK*. Se analizează modul în care această metodă rezolvă problema staţiei ascunse:

- Staţia A ascultă mediul şi nu detectează prezenţa unui semnal începe să transmită
- Staţia C face exact acelaşi lucru şi începe şi ea să transmită
- Se produce o coliziune între cele două cadre şi niciunul din ele nu ajunge la staţia B, deci staţia B nu trimite *ACK* nici staţiei A, nici staţiei C
- Ambele staţii aşteaptă un timp predefinit în care aşteaptă *ACK*. Dacă timpul de aşteptare expiră ambele staţii vor aştepta un timp aleatoriu numit DIFS (*Distributed Interframe Spacing*) înainte sa transmită din nou.

Atenţie! *deoarece în* wireless, *pentru fiecare cadru trimis, se aşteaptă un* ACK, *banda efectivă de care se dispune, este de la început înjumătăţită.*

O altă metodă de acces la mediu prevăzută în CSMA/CA presupune folosirea unor mesaje speciale de tip RTS (*request to send*) şi CTS (*clear to send*). Folosind acest mecanism, o staţie întreabă AP-ul dacă mediul de transmisie este liber folosind un mesaj de tip RTS. Acest cadru ajunge doar la AP. AP-ul, fiind punctul central al reţelei, nu are problema staţiei ascunse. Dacă mediul este liber, AP-ul trimite un CTS în care specifică staţia ce a obţinut permisia de a transmite. Mesajul de tip CST ajunge la toate staţiile din reţea, nu doar la cea care a cerut acces la mediu prin mesajul RTS anterior. Astfel staţia ce dorea să transmită va primi acces, iar celelalte staţii vor considera mediul ocupat de aceasta.

Material cu drept de autor

5.1.6.3 Canale de comunicație

După cum se poate observa până în acest punct, deşi CSMA/CA prevede o metodă de comunicare funcțională, totuşi nu oferă o soluție ca două stații să poată transmite în acelaşi timp, pe aceeaşi frecvență. S-ar obține o coliziune urmată de retransmiterea pachetului. Deci, cum se pot obține două rețele *wireless*, care să funcționeze în aceeaşi banda ISM, şi care să nu producă o coliziune sau să interfereze una cu cealaltă. Răspunsul rezidă în analiza lățimii de bandă care este folosită într-o transmisie *wireless*. Lățimea de bandă este până la urma o plaja de frecvențe în care se transmite un semnal.

Se va lua ca exemplu standardul 802.11g. Pentru a putea transmite 54 Mbps, folosind OFDM şi tehnici de transmisie în spectru împrăştiat, este nevoie de o plajă de frecvențe de doar 22 Hz. Dar banda ISM de 2,4 GHz se întinde între 2,401 şi 2,473 Ghz (o plajă de 72 Hz). Deci se pot transmite simultan în această banda ISM, 3 (72/22) fluxuri *wireless* separate, care funcționează fiecare într-o bandă separată de 22 Hz.

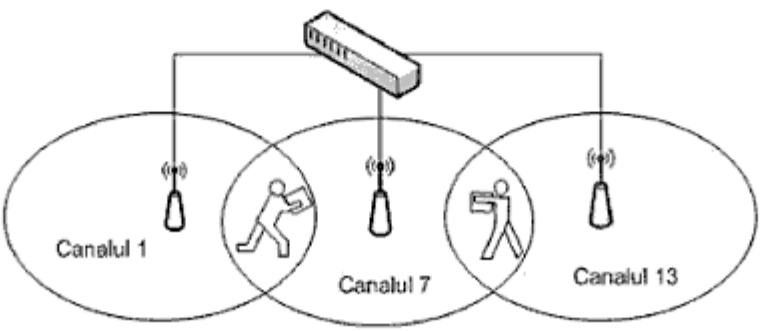
Aceste benzi de 22 Hz, se numesc canale. Folosirea mai multor canale, face posibilă coexistența a două rețele *wireless* care funcționează în aceeaşi bandă ISM, în acelaşi domeniu de propagare. Standardul 802.11g specifică existența mai multor astfel de canale care sunt spațiate la doar 5 MHz unul de celălalt şi care se suprapun pe o bandă de 17 MHz. De ce sunt prevăzute mai multe canale care interferează unul cu celălalt, în loc de trei canale complet independente? Pentru că se doreşte posibilitate reglajului fin în cazul unei interferențe de interval mic la unul din capetele unui canal.

Lărgimea benzii ISM diferă între standardul din America şi cel din Europa. În timp ce în America se specifică intervalul 2,401 - 2,473 *Ghz*, în Europa se precizează plaja de 2,401 GHz până la 2,483 GHz. Cele 3 canale ce nu interferează sunt 1,6 şi 11 în America şi 1, 7 şi 13 în Europa.

În anul 2003 IEEE a publicat standardul 802.11h, care a fost încorporat în 802.11 în 2007. Acest amendament adus standardului original introduce şi canale suplimentare în benzile UNII pentru rețelele 802.11a. În prezent, se beneficiază de 23 de canale în Statele Unite şi de 19 în Europa.

5.1.6.4 Roaming

Colocalizarea mai multor rețele poate fi folosită şi în alt scop: mărirea acoperirii şi benzii unei rețele *wireless* prin adăugarea de *access point*-uri. Pentru a realiza acest deziderat, nu este suficient să adăugam mai multe AP-uri la rețea aşa cum punem mai multe becuri pentru a face mai multă lumină. În momentul în care apar mai multe *access point*-uri, automat se creează mai multe rețele *wireless*. În consecință, vor apărea coliziuni între ele dacă nu folosim canale disjuncte.



5-8: Roaming

Material cu drept de autor

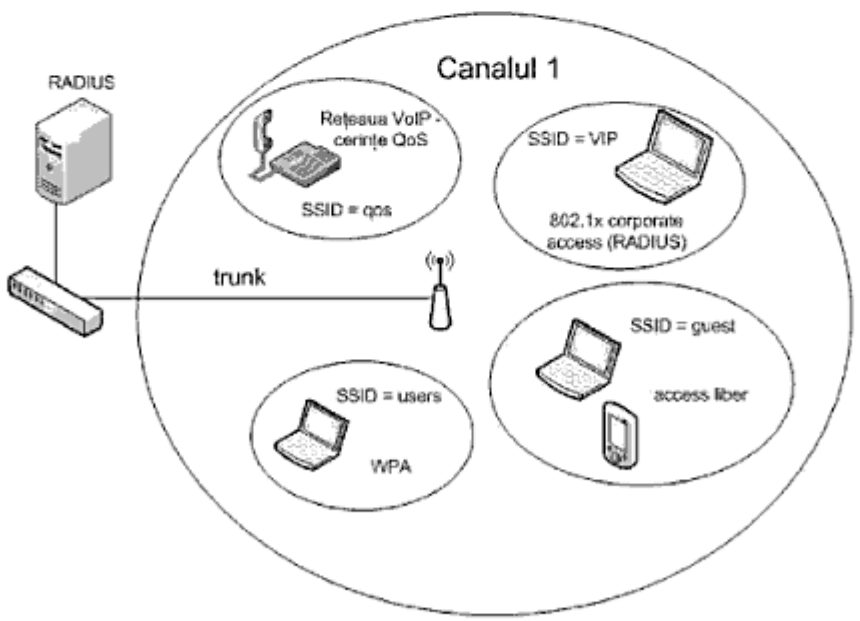
Topologia din figură poartă numele de ESS (*Extended Service Set*) şi funcţionează astfel: AP-urile sunt conectate în acelaşi reţea *Ethernet* şi funcţionează respectiv pe 3 canale disjuncte (1-7-13). Astfel, când o staţie intră în raza de acoperire a reţelei, ea va fi asociată AP-ului cu semnalul cel mai puternic din acea zona. Aici se aplică funcţia de reasociere a AP-urilor. Dacă o persoană cu un PDA traversează de la stânga la dreapta reţeaua, PDA-ul va fi asociat pe rând fiecăruia dintre AP-uri. Spre exemplu când semnalul de la *access point*-ul 1 devine prea slab, adaptorul va fi intrat deja în raza *access point*-ului 2 şi va fi asociat acestuia fără pierderea conexiunii la reţea. Acest serviciu, ca şi la telefonia mobilă, poartă numele de **roaming**.

Folosind o topologie ESS, nu se extinde doar raza reţelei, ci şi se oferă mai multă bandă clienţilor mobili. Aceasta pentru că fiecare AP oferă clienţilor lui până la 54 Mbps, deci în total ele vor constitui o reţea *wireless* cu o bandă de 3x54 Mbps = 162 Mbps.

5.1.6.5 VLAN-uri

VLAN-urile se configurează în general pe switchuri pentru a putea obţine mai multe domenii de *broadcast*. Aceste *VLAN*-uri se pot extinde şi în reţelele *wireless* prin maparea unui SSID diferit pentru fiecare VLAN dorit . Prin aceasta operaţie se obţine o virtualizare a *access point*-ului, acesta apărând ca mai multe reţele diferite. Singura limitare este ca toate reţele trebuie să funcţioneze pe acelaşi canal căci un transmiţător *wireless* nu poate genera trafic pe mai multe canale în acelaşi timp. Bineînţeles că nu orice AP va putea să suporte *VLAN*-uri. De fapt, singurul lucru de care are nevoie AP-ul, este suport pentru protocolul 802.1q, pentru a putea realiza un *trunk*[1] cu switchul în care este legat. Unele AP-uri suportă chiar si tehnici de *QoS* (802.1p[2]) pentru a putea clasifica importanţa traficului din fiecare VLAN.

Un motiv important pentru care s-ar dori existenţa *VLAN*-urilor pe AP-uri este separarea reţelei *wireless* după diferite privilegii de securitate (*guest*, *user*, VIP).



5-9: VLAN-uri

[1] A se vedea capitolul 2
[2] http://en.wikipedia.org/wiki/802.1p

Material cu drept de autor

5.1.7 Securitatea *wireless*

5.1.7.1 SSID *broadcast*

Fiecare rețea *wireless* este identificată printr-un nume: așa numitul SSID (*Service Set Identifier*). Pentru ca o stație să se poată asocia la o rețea, ea trebuie să cunoască *SSID*-ul rețelei respective. În mod normal, *access point*-urile anunță periodic SSID-ul rețelei pentru ca o stație să verifice dacă nu cumva dorește să se asocieze rețelei respective. Acest proces de anunțare se numește SSID *broadcast*. Cunoașterea SSID-ului de către o stație atacatoare poate fi un risc de securitate. Toate AP-urile oferă o setare ce permite eliminarea SSID *broadcast*, forțând astfel o asociere activă (stația trebuie să specifice manual rețeaua la care dorește să se conecteze).

Deși eliminarea SSID *broadcast* este considerată în unele documentații ca fiind o măsura de securitate, această afirmație este eronată. Protocolul 802.11 conține în specificații mesaje speciale numite *beacons*. Aceste mesaje sunt folosite de către AP ca un mecanism de verificare periodică a faptului că o stație este încă activă. *Beacon*-urile sunt văzute de toate stațiile din rețea și conțin SSID-ul rețelei. Deci eliminarea SSID *broadcast* este o măsura complet inutilă de securitate, odată ce SSID-ul poate fi aflat de atacator din aceste *beacon*-uri ce nu pot fi dezactivate.

Trebuie spus totuși că există unele routere care permit eliminarea completă a SSID-ului, atât din mesajele de *broadcast*, cât și din *beacon*-uri.

5.1.7.2 Filtrare bazată pe adresa MAC

O metodă foarte des folosită în rețelele mici este *MAC Filtering*. Aceasta presupune configurarea AP-ului astfel încât să permită accesul doar anumitor interfețe *wireless*, pe baza adresei MAC. Spre exemplu, dacă în rețeaua voastră vreți să conectați doar aceleași cinci stații, puteți configura AP-ul să accepte doar interfețele care au una dintre cele 5 adrese *MAC*. Nici această abordare nu e mult mai sigură: adresele *MAC* acceptate pot fi aflate prin captura traficului între ele și AP. Apoi un hacker poate schimba adresa *MAC* a interfeței proprii cu unul dintre *MAC*-urile capturate.

Metodele prezentate mai sus, deși nu sunt foarte sigure, sunt foarte des folosite. Aceasta nu din lipsa de cunoștințe sau ignoranță. Când vine vorba despre securitate, înaintea luării oricărei măsuri, trebuie realizată o evaluare a riscului. Dacă aveți de realizat o rețea *wireless* într-o centrală nucleară, atunci merită asigurat un maxim de securitate. Dacă însă aveți o rețea acasă și protejați o conexiune Internet, atunci măsurile nu au de ce sa fie foarte drastice.

5.1.7.3 Securitatea WEP

Protocolul 802.11 include și o specificație pentru un mecanism de securizare mai avansat: WEP (*Wired Equivalent Privacy*). După cum spune și numele, reprezintă un mecanism specializat ce, teoretic, ar trebui să confere un nivel destul de ridicat de securitate.

WEP asigură criptarea traficului din rețea printr-un algoritm de criptare simetrică denumit RC4. Pentru a securiza o rețea prin acest mecanism, o cheie specifică rețelei (nu este același lucru cu SSID) este cunoscută de către toate nodurile cărora li se permite asocierea. În momentul în care o stație dorește să se asocieze AP-ului, acesta verifică dacă stația cunoaște cheia secretă printr-o secvență de *challenge*: trimite un mesaj aleator pe care stația trebuie să-l returneze criptat. AP-ul va decripta apoi mesajul și va verifica dacă este chiar mesajul pe care el l-a trimis inițial. Acest pas suplimentar la asociere poartă numele de autentificare. Din acest moment stația poate comunica în rețea, însă cadrele comunicate vor fi toate criptate cu cheia rețelei.

Material cu drept de autor

Din nefericire, oricât de puternică ar fi criptarea şi oricât ar fi cheia de lungă, faptul că aceasta este statică este o mare vulnerabilitate. Un hacker poate intercepta secvența de autentificare şi, cunoscând algoritmul, va încerca să afle cheia. Din moment ce aceasta se va schimba foarte rar, are foarte mult timp la dispoziție să o facă.

Fiind vorba de tehnologie relativ recentă, s-au făcut multe studii în legătură cu puterea *WEP*-ului, descoperindu-se astfel o serie de vulnerabilități. Totul a culminat cu spargerea algoritmului în 2001, iar acum există publicate pe Internet aplicații care pot descoperi foarte repede cheia WEP pe baza pachetelor interceptate din rețea.

5.1.7.4 Securitatea WPA/WPA2

Wi-Fi Alliance este un consorțiu format din peste 300 dintre cele mai mari companii din domeniul IT, având ca scop promovarea şi dezvoltarea tehnologiilor *wireless*. În 2003 *Wi-Fi Alliance* a propus un nou protocol pentru securitatea rețelelor: WPA (*Wi-Fi Protected Access*) ce a devenit standardul de securitate pentru majoritatea echipamentelor 802.11.

WPA implementează un subset al specificațiilor de securitate *wireless* prezente în standardul 802.11i şi reprezintă răspunsul industriei la şocul spargerii *WEP*-ului, venind cu soluții pentru cele mai importante vulnerabilități descoperite acestuia din urmă. Cea mai importantă îmbunătățire adusă WPA-ului este folosirea unui protocol de schimbare dinamică a cheii de criptare pe durata conexiunii: TKIP (*Temporary Key Integrity Protocol*). Acesta vine să rezolve problema WEP legată de posibilitatea facilă de recuperare a cheii. WPA funcționează în două moduri: **personal** şi **enterprise**.

În prezent, WPA a ajuns la cea de-a doua versiune. În iunie 2004 IEEE a omologat standardul 802.11i, venit să amendeze standardul 802.11 cu privire la securitatea în rețelele *wireless*. WPA2 implementează setul de specificații obligatorii prezent în standardul 802.11i, păstrând şi compatibilitatea cu prima versiune a protocolului (WPA). Recomandările actuale în privința securității rețelelor fără fir sunt folosirea unui sistem WPA2 Personal (*AES PreShared Key*) pentru utilizatorii casnici şi WPA2 Enterprise împreună cu un server RADIUS pentru mediul de afaceri.

În pofida existenței acestor tehnici avansate de securitate, un studiu efectuat în iunie 2007 în Londra, oraşul cu cel mai mare număr de rețele fără fir din lume (7130 de puncte de acces), a arătat că 19% din rețelele fără fir nu aveau implementat niciun fel de sistem de protecție a rețelei. Dintre cele care ofereau un oarecare nivel de securitate, 48% erau protejate cu WEP.

Atenție! *Dacă parola pe care o folosiți pentru securizarea rețelei este slabă, nu contează ce algoritm sau ce standard de securitatea folosiți: rețeaua va putea fi spartă printr-un atac de tip brute force! De aceea se recomandă citirea unui ghid*[1] *ce prezintă cele mai bune practici în alegerea unei parole, înainte de a configura securitatea unei rețele.*

5.2 Configurarea unei rețele wireless *în Linux – configurări de bază*

5.2.1 De ce *wireless* pe Linux?

Pe parcursul ultimilor 5 ani, s-a investit destul de mult efort atât din partea comunității cât şi din partea a diferite organizații (*Linux Foundation*, *WiFi Alliance*, *DELL*) pentru a îmbunătăți şi a populariza Linux ca o platformă mobilă pentru utilizatorul final. Prezentându-se în lumea utilizării sistemelor de operare ca o platformă cu pretenții, odată cu apariția nevoii de mobilitate pe piața IT, Linux a obținut suport pentru stiva 802.11 şi a pătruns şi în lumea dispozitivelor *embedded*, dezvoltând platforme împreună cu Texas Instruments şi Intel.

[1] http://www.ou.edu/committees/itc/policy/Guidelines_for_Passwords.html

Material cu drept de autor

În prezent, s-au făcut progrese reale pentru oferirea unui suport cât mai bun pentru drivere *wireless*, astfel că, în prezent, Linux oferă atât soluții de rețele `adhoc`, infrastructură de dimensiuni mici și medii cât și soluții *enterprise*.

Unul din proiectele interesante care își propune să mărească gama de suport *wireless* în Linux este *NDISwrapper*[1]. Pentru că mulți producători nu eliberează și drivere Linux pentru plăcile *wireless* livrate, *NDISwrapper* face posibilă instalarea de drivere *wireless* scrise în Windows API pe o platformă Linux. Mai multe informații se pot găsi pe pagina web a proiectului.

5.2.2 Configurări de bază

Se vor prezenta în continuare, atât modul de configurare ale unei rețele *wireless* `adhoc` și infrastructură, cât și modalități de securizare pentru acestea. Deși Ubuntu oferă suport pentru configurarea în mediul grafic, pentru exemplele de mai jos se va preferă utilizarea linie de comandă. Motivul pentru această alegere este lipsa de suport pe termen lung pentru clientul de *wireless* pe care Ubuntu îl conține și instabilitatea pe care *GUI*-ul o are pentru unele drivere *wireless*. Folosind linia de comandă, există siguranța unei soluții independente de platformă și care va putea fi aplicată pe orice distribuție de Linux

Configurarea unei rețele (atât *ad hoc* cât și infrastructură) se poate face în două moduri:

- temporar - prin intermediul unor comenzi ce se aplică în momentul în care sunt introduse. Ele afectează *software*-ul ce rulează în RAM, deci configurările nu vor fi persistente la următoarea pornire a sistemului.
- permanent - prin editarea fișierului de configurare `/etc/network/interfaces` și restartarea serviciului de rețea. Fișierul este încărcat la fiecare repornire a sistemului.

5.2.2.1 Pregătirea interfeței de rețea

Un pachet esențial pentru acest capitol, care va fi folosit în toate configurațiile prezentate, este `wireless-tools`[2]. Acesta este inclus în Ubuntu 8.04, iar în cazul unei versiuni mai vechi de Ubuntu se poate instala ușor folosind utilitarul apt:

```
waterstmyr:~$ sudo apt-get install wireless-tools
```

Acest pachet conține următoarele utilitare:

- `iwconfig` este un utilitar asemănător `ifconfig` cu ajutorul căruia se pot configura parametrii de bază ai unei legături *wireless*.
- `iwlist` oferă posibilitatea de scanare în linie de comandă pentru găsirea rețelelor *wireless*.
- `iwevent` permite monitorizarea interfeței *wireless* și raportarea unui eveniment de asociere cu o rețea *wireless* sau încheierea unei scanări.
- `iwspy` afișează parametrii de putere și de calitate ai legăturii *wireless*
- `iwpriv` permite modificarea unor parametrii specifici fiecărui *driver wireless*

În multe cazuri, dificultatea realizării unei rețele *wireless* pe Linux nu constă în configurările propriu-zise, ci în diferite incompatibilități cu drivere, cu diferite aplicații activate în mod implicit în Ubuntu, sau chiar cu plăci *wireless* pentru care nu există încă suport în Ubuntu. Acest subcapitol a fost creat pentru a elimina unele probleme ce ar putea îngreuna realizarea unei rețele *wireless* sau ar putea împiedica setarea unor parametrii de bază.

Se vor parcurge câțiva pași premergători, ce vor asigura buna funcționare a configurațiilor din acest capitol.

[1] http://ndiswrapper.sourceforge.net/joomla/
[2] http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html

Material cu drept de autor

1. Oprirea interfeței *Ethernet*.

Deși nu este un caz des întâlnit, unele drivere *wireless* nu permit configurarea rețelei dacă detectează conectivitate pe o legătura *Ethernet*, deci înainte de a configura rețeaua *ad hoc* trebuie decuplat cablul *Ethernet*.

2. Oprirea *firewall*-ului

Trebuie asigurat că nu aveți niciun *firewall* (precum `firestarter`), instalat și rulând în sistem. În Ubuntu există un *firewall* implicit numit `iptables`. Pentru a fi siguri că rețeaua nu va fi blocată de o regulă introdusă accidental sau o regulă ce blochează mai mult decât ar trebui să o facă, se recomandă ștergerea tabelei de filtrare a `iptables` astfel:

```
waters@myr:~$ sudo iptables -F
```

3. Oprirea *NetworkManager*-ului

Aplicația din Ubuntu ce se ocupă cu operațiile clientului de *wireless* se numește *NetworkManager*. Aceasta va trebui oprită înainte de a face orice configurare. Oprirea programului se poate face prin apelarea scriptului numit `25NetworkManager` cu parametrul `stop`.

```
waters@myr:~$ sudo /etc/dbus-1/event.d/25NetworkManager stop
 * Stopping network connection manager NetworkManager                [ OK ]
```

4. Compatibilității hardware (**doar** pentru realizarea unei rețele *ad hoc*)

Se recomandă verificarea pe lista oficială de pe pagina web Ubuntu[1] dacă placa *wireless* suportă modul *ad hoc* pe Ubuntu.

5. Verificarea driverului

Este posibil să apară probleme cu *driver*-ul instalat implicit de Ubuntu sau recunoașterea plăcii *wireless* de către sistemul de operare. Pentru a putea verifica detectarea plăcii *wireless*, se vor folosi comenzile:

- `lspci` pentru plăcile *wireless* ce funcționează pe port PCI.
- `lsusb` pentru plăcile *wireless* ce funcționează pe port USB (adaptoare *wireless*).

```
waters@myr:~$ lspci
[...]
04:00.0 Network controller: Intel Corporation PRO/Wireless 3945ABG Network Connection
(rev 02)
```

Dacă placa *wireless* nu apare în *output*, cel mai probabil Ubuntu nu a avut integrat un driver pentru placa de rețea. În acest caz trebuie căutat pe *site*-ul producătorului un driver *wireless*, iar dacă nu există driver pentru Linux, puteți folosi soluția *NDISwrapper* pentru a instala un driver de Windows.

5.2.2.2 Configurarea unei rețele *ad hoc*

> ***Notă**: toate configurările ce urmează trebuie realizate pe toate stațiile din rețeaua ad hoc.*

După cum s-a precizat, configurarea unei stații se poate face temporar sau permanent. Se vor urmări ambele abordări în cele ce urmează.

5.2.2.2.1 Configurarea temporară

Pentru început, trebuie aflat numele interfeței *wireless* de rețea. Se va folosi comanda `iwconfig` fără niciun parametru pentru a afișa interfețele active în sistem.

```
waters@myr:~$ iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.
```

[1] https://help.ubuntu.com/community/WifiDocs/WirelessCardsSupported

Material cu drept de autor

```
wlan0   IEEE 802.11g  ESSID:""  Nickname:""
          Mode: auto  Frequency:2.412 GHz  Cell: Not-Associated
          Tx-Power=27 dBm
          Retry min limit:7   RTS thr:off   Fragment thr=2346 B
          Power Management:off
          Link Quality:0  Signal level:0  Noise level:0
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

Notă: *sintaxa comenzii* `iwconfig` *este:* ***iwconfig*** *<interfața_*wireless*> <parametru_rețea> <valoarea_parametrului>*

Dacă interfața nu apare în rezultatul comenzii `iwconfig`, numele cu care sistemul identifică această componentă (`wifi0`, `wlan0`, `wireless0`) poate fi aflat din consultarea fișierului `/proc/net/dev`.

```
waters@myr:~$ cat /proc/net/dev
Inter-|   Receive                                                |  Transmit
 face |bytes    packets errs drop fifo frame compressed multicast|bytes    packets errs
drop fifo colls carrier compressed
lo:   14016     184    0    0    0     0          0         0    14016     184    0    0    0     0       0          0
eth1: 1461444   10683  0    0    0     0          0         0  1034216    9140    0    0    0     0       0          0
```

Următoarele comenzi vor trebui introduse în ordinea în care sunt listate mai jos. Trebuie avut grijă cu secvențierea comenzilor căci unele comenzi pot afecta setările realizate anterior.

Pentru a evita probleme de configurare ce ar putea apărea din cauza *driver*-ului, se va opri interfața *wireless* înainte de aplicarea parametrilor de rețea:

```
waters@myr:~$ sudo ifconfig wlan0 down
```

Specificarea tipului de rețea *wireless* se face cu ajutorul parametrului mode al comenzii `iwconfig`.

```
waters@myr:~$ sudo  iwconfig wlan0 mode adhoc
```

Parametrul `mode` are cea mai mare prioritate în configurație și trebuie introdus întotdeauna primul în secvența de comenzi de configurare. Valorile sale relevante sunt:

- *master* - această valoare desemnează interfața ca fiind una de AP. Se va folosi numai dacă se dorește realizarea unui server Linux care să poată funcționa ca *Access Point*. Se recomandă consultarea *WiFi-Docs*[1] pentru posibilitatea setării plăcii în acest mod.
- *managed* – acest mod se folosește când se dorește conectarea la un AP într-o rețea tip infrastructură
- *ad hoc* – specifică opțiunea unei rețele *ad hoc*

În continuare, trebuie configurat SSID-ul rețelei *ad hoc* pe fiecare dintre stații.

```
waters@myr:~$ sudo iwconfig wlan0 essid my_network
```

Pentru unele adaptoare *wireless* este necesară și configurarea unui canal de comunicare.

```
waters@myr:~$ sudo iwconfig wlan0 channel 1
```

Într-o rețea `adhoc` se pot folosi și alți parametrii opționali ai comenzii `iwconfig` pentru a seta *bit-rate*-ul, puterea de transmisie sau parametrii de securitate.

Până la acest pas, a fost configurată o rețea *wireless* ce oferă conectivitate de nivel 2. Pentru a putea asigura comunicare de nivel 7 trebuie configurați, mai întâi, parametrii de nivel 3: adresele IP (de asigurarea conectivității 4-7 se ocupa stiva TCP/IP și sistemul de operare).

[1] https://help.ubuntu.com/community/WifiDocs/MasterMode

Material cu drept de autor

***Notă**: În general în lumea reţelelor, administratorul de reţea se ocupă cu configurările de nivel 1,2,3 şi 4; mai puţin cu nivelul aplicaţie.*

Setarea adresei IP se face folosind comanda `ifconfig`:

```
waters@myr:~$ sudo ifconfig wlan0 192.168.1.1 netmask 255.255.255.0 broadcast
192.168.1.255 up
```

***Atenţie!** toate staţiile din reţeaua ad hoc trebuie să se afle în acelaşi subnet pentru ca reţeaua să poată funcţiona corect. Se poate folosi utilitarul `ping` pentru verificarea legăturii de nivel 3 şi SSH pentru testarea legăturii de nivel 7.*

5.2.2.2.2 Configurarea permanentă

După cum s-a mai precizat, pentru a realiza o configuraţie permanentă trebuie editat fişierul text `/etc/network/interfaces` cu parametrii reţelei.

***Notă:** în general pe Linux toate configurările permanente se fac în fişiere text. Chiar şi atunci când se folosesc utilitare în mod grafic, acestea operează modificări în fişiere text.*

Deşi sintaxa este puţin schimbată, se configurează aceiaşi parametrii precum în cazul configurării temporare:

```
auto wlan0
iface wlan0 inet static
wireless-mode adhoc
wireless-channel 4
wireless-essid my_network
address 192.168.0.2
netmask 255.255.255.0
```

***Atenţie!** ordinea parametrilor de configurare contează.*

Pentru a putea testa aplicarea corecta a parametrilor introduşi în fişierul de configurare, trebuie forţată reinterpretarea fişierului `/etc/network/interfaces`. Acest lucru se poate face prin repornirea serviciului de reţea:

```
waters@myr:~$ sudo /etc/init.d/networking restart
```

Rezultatul comenzii `iwconfig` va reflecta parametrii aplicaţi:

```
waters@myr:~$ iwconfig wlan0

wlan0     IEEE 802.11g  ESSID:"my_network"  Nickname:""
          Mode:Adhoc  Frequency:2.427 GHz  Cell: 00:19:E0:84:DD:4A
          Bit Rate=54 Mb/s   Tx-Power=27 dBm
          Retry min limit:7   RTS thr:off   Fragment thr=2346 B
          Power Management:off
          Link Quality=85/100  Signal level=-48 dBm  Noise level=-82 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

5.2.2.3 Configurarea unei reţele de tip infrastructură

Ca şi în cazul reţelei *ad hoc* configurarea se poate face temporar, prin comenzi `iwconfig`, sau permanent, prin editarea fişierului de configurare. Pentru ambele tipuri, vor trebui mai întâi parcurse etapele de pregătire ale interfeţei de reţea descrise anterior.

Material cu drept de autor

5.2.2.3.1 Configurarea temporară

Pentru o rețea de tip infrastructură va trebui setat modul *managed*, SSID-ul rețelei la care se dorește conectarea și canalul de comunicație folosit de rețea. SSID-ul și canalul de comunicație identifică în mod unic o rețea *wireless*, deci trebuie setate pe client la aceeași valoare la care au fost configurate și pe AP.

Dacă se dorește apartenența la o rețea fără fir, trebuie aflat cumva SSID-ul rețelei la care trebuie făcută asocierea. Pachetul `wireless-tools` include un utilitar numit `iwlist`, folosit pentru scanarea mediului și aflarea SSID-ului rețelelor care se află în raza receptorului *wireless* local. Pentru a porni scanarea, se va apela comanda `iwlist` astfel:

```
waters@myr:~$ sudo iwlist wlan0 scanning

wlan0     Scan completed :
          Cell 01 - Address: 00:1B:FC:60:D7:8D
                    ESSID:"YoGi"
                    Mode:Master
                    Channel:1
                    Frequency:2.412 GHz (Channel 1)
                    Quality=71/100  Signal level=-63 dBm  Noise level=-75 dBm
                    Encryption key:on
                    IE: WPA Version 1
                    Group Cipher : TKIP
                    Pairwise Ciphers (2) : CCMP TKIP
                    Authentication Suites (1) : PSK
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 18 Mb/s
                    24 Mb/s; 36 Mb/s; 54 Mb/s; 6 Mb/s; 9 Mb/s
                    12 Mb/s; 48 Mb/s ; 54 Mb/s
                    Extra:tsf=000000ca49a2cfbc

          Cell 02 - Address: 00:19:E0:84:DD:4A
                    ESSID:"guest"
                    Mode:Master
                    Channel:11
                    Frequency:2.412 GHz (Channel 1)
                    Quality=79/100  Signal level=-55 dBm  Noise level=-75 dBm
                    Encryption key:off
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s
                    Extra:tsf=0000000001909000
```

Analizând rezultatul comenzii, se poate observa existența a două rețele de tip infrastructură (identificate astfel după parametrul `mode` Master) la care se poate încerca conectarea:

- Rețeaua cu SSID-ul YoGi
  - folosește frecvența de 2.4 Ghz și canalul 1
  - Adresa MAC prezentă în directive "Cell:" este adresa fizică a AP-ului și poartă numele de BSSID (*Basic Service Set Identifier*) într-o rețea infrastructură.
  - Analizând bit rate-ul posibil pe care AP-ul îl oferă, se poate concluziona faptul că AP-ul folosește protocolul 802.11g deoarece 802.11b, deși funcționează la aceeași frecvență, nu are o rată teoretică de funcționare mai mare de 11 Mbps
  - folosește autentificare și criptare pe baza protocolului WPA TKIP (chei schimbate în mod dinamic).
- Rețeaua cu SSID-ul *Guest*
  - folosește frecvența de 2.4 Ghz și canalul 11
  - Adresa MAC prezentă în directive "Cell:" este adresa fizică a AP-ului și poartă numele de BSSID (Basic Service Set Identifier) într-o rețea infrastructură.
  - Urmărind rezultatul ratei posibile afișate (bit rate), se poate identifica rețeaua ca fiind 802.11b (viteză maximă 11 Mbps)
  - rețeaua nu folosește asociere securizată, fiind de tip OPEN (Encryption off). Atenție, acest lucru nu înseamna ca se va putea realiza întotdeauna asocierea la rețea. Chiar daca rețeaua apare ca fiind OPEN la scanare, pot fi implementate politici de filtrare după adresa MAC pe AP, care să permită doar anumitor stații să se asocieze rețelei.

Se va configura în continuare asocierea la rețeaua *guest* prin:

- oprirea temporară a interfeței de rețea cât timp se aplică noile configurații
- setarea modului *managed*
- setarea SSID-ului și canalului corespunzător rețelei *guest*
- deschiderea interfeței *wireless*

Material cu drept de autor

Comenzile pentru efectuarea operaţiilor de mai sus sunt după cum urmează:

```
waters@myr:~$ sudo  ifconfig wlan0 down
waters@myr:~$ sudo  iwconfig wlan0 mode managed
waters@myr:~$ sudo  iwconfig wlan0 essid guest
waters@myr:~$ sudo  iwconfig wlan0 channel 11
waters@myr:~$ sudo  ifconfig wlan0 up
```

Configurarea de nivel 2 este completă şi se poate verifica prin `iwconfig`:

```
waters@myr:~$ iwconfig wlan0

wlan0     IEEE 802.11g  ESSID:"guest"  Nickname:""
          Mode:Managed  Frequency:2.462 GHz  Access Point: 00:19:E0:84:DD:4A
          Bit Rate=54 Mb/s   Tx-Power=27 dBm
          Retry min limit:7   RTS thr:off   Fragment thr=2346 B
          Power Management:off
          Link Quality=68/100  Signal level=-65 dBm  Noise level=-75 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

Observaţie: *deoarece încă nu s-a realizat asocierea la reţeaua 802.11b cu SSID guest, bit-rate-ul clientului este încă setat pe valoarea implicită de 54 Mbps. Cum standardul 802.11g este cel mai întâlnit în reţele fără fir, majoritatea clienţilor vor avea caracteristicile implicite ale acestui protocol.*

Mai rămâne de configurat doar asignarea unui IP pe interfaţa *wireless* pentru a putea comunica la nivel 3 în reţea. Una din diferenţele importante dintre o reţea *ad hoc* şi una infrastructură este una de administrare. Într-o reţea *ad hoc* nu există administrator şi de aceea fiecare utilizator trebuie să seteze IP-ul static pe interfaţa de reţea, având grijă ca toate IP-urile să se afle în acelaşi *subnet*. Reţeaua de tip infrastructură va presupune un minim de administrare şi mai mult, existenţa unei metode centralizate de setare a adreselor IP pe fiecare staţie, fără intervenţia utilizatorilor: un server DHCP. În cazul în care AP-ul oferă funcţionalitatea unui server DHCP, intervalul din care serverul DHCP oferă adrese IP poate fi configurat din interfaţa grafica a AP-ului.

5-10: configurare serverului DHCP

Deci, se poate finaliza configurarea clientului *wireless* prin efectuarea unei cereri DHCP pe interfaţa `wlan0` cu ajutorul comenzii `dhclient`:

```
waters@myr:~$ sudo dhclient wlan0

Internet Systems Consortium DHCP Client V3.0.6
Copyright 2004-2007 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/

Listening on LPF/wlan0/00:18:de:b9:ac:da
Sending on   LPF/wlan0/00:18:de:b9:ac:da
Sending on   Socket/fallback
DHCPDISCOVER on wlan0 to 255.255.255.255 port 67 interval 5
DHCPOFFER of 192.168.2.105 from 192.168.2.1
DHCPREQUEST of 192.168.2.105 on wlan0 to 255.255.255.255 port 67
DHCPACK of 192.168.2.105 from 192.168.2.1
bound to 192.168.2.105 -- renewal in 48000 seconds.
```

Material cu drept de autor

5.2.2.3.2 Configurarea permanentă

Pentru o configurație persistentă, introducem directivele specificate anterior, în fișierul `/etc/network/interfaces`

```
auto wlan0
iface wlan0 inet dhcp
wireless-mode managed
wireless-channel 11
wireless-essid guest
```

Prin realizarea repornirii serviciului de rețea, se vor aplica parametrii `mode`, `channel` si `SSID`, iar apoi se va face o cerere DHCP pentru configurarea parametrilor de nivel 3.

```
waters@myr:~$ sudo /etc/init.d/networking restart

 * Reconfiguring network interfaces...
Internet Systems Consortium DHCP Client V3.0.6
Copyright 2004-2007 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/

Listening on LPF/wlan0/00:18:de:b9:ac:da
Sending on   LPF/wlan0/00:18:de:b9:ac:da
Sending on   Socket/fallback
DHCPDISCOVER on wlan0 to 255.255.255.255 port 67 interval 3
DHCPOFFER of 192.168.2.107 from 192.168.2.1
DHCPREQUEST of 192.168.2.107 on wlan0 to 255.255.255.255 port 67
DHCPACK of 192.168.2.107 from 192.168.2.1
bound to 192.168.2.107 -- renewal in 53306 seconds.
                                                          [ OK ]
```

5.2.2.3.3 Testarea rețelei

În continuare se va analiza o problemă de conectivitate ce poate apărea după configurarea unei rețele de tip infrastructură. Se va presupune că administratorul efectuează un `ping` către adresa `www.google.com`, iar comanda întoarce mesajul `ICMP: Destination Unreachable`. În mod evident există o problema de conectivitate în rețea. Pentru a o rezolva, se vor parcurge următorii pași:

1. Verificarea corectă a asocierii la rețeaua 802.11b

După asociere se poate verifica sincronizarea parametrilor între clientul *wireless* și rețeaua 802.11b, printre care adresa MAC a AP-ului cu care s-a făcut asocierea și rata de transfer ce a fost negociată la maximul posibil al rețelei: 11Mpbs.

```
waters@myr:~$ iwconfig wlan0

wlan0     IEEE 802.11g  ESSID:"guest"  Nickname:""
          Mode:Managed  Frequency:2.462 GHz  Access Point: 00:19:E0:84:DD:4A
          Bit Rate=11 Mb/s   Tx-Power=27 dBm
          Retry min limit:7   RTS thr:off   Fragment thr=2346 B
          Power Management:off
          Link Quality=93/100  Signal level=-37 dBm  Noise level=-75 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

2. Verificarea conectivității la *gateway*.

O cererea DHCP efectuată configurează mai mult decât adresa IP pe interfața *wireless*; aceasta introduce și o ruta implicită (rută *default*) către adresa IP a *gateway*-ului și oferă stației și o adresă validă a unui server DNS. Existența rutei implicite se verifică prin comanda `route`:

```
waters@myr:~$ route -n

Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.2.0     0.0.0.0         255.255.255.0   U     0      0        0 wlan0
0.0.0.0         192.168.2.1     0.0.0.0         UG    100    0        0 wlan0
```

Rezultatul comenzii conține o rută implicită către adresa 192.168.2.1. După ce s-a aflat adresa IP a *gateway*-ului, se poate testa conectivitatea cu acesta prin comanda `ping`:

Material cu drept de autor

```
waters@myr:~$ ping 192.168.2.1 -c 3
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=64 time=1.75 ms
64 bytes from 192.168.2.1: icmp_seq=2 ttl=64 time=1.64 ms
64 bytes from 192.168.2.1: icmp_seq=3 ttl=64 time=2.19 ms

--- 192.168.2.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 1.640/1.865/2.199/0.240 ms
```

3. Verificarea existenţei unei adrese IP pentru serverul de nume (DNS)

Dacă se ajunge în acest pas, se poate considera ca accesul în reţeaua locală este lipsit de probleme, odată ce se poate da `ping` în adresa de *gateway*. Singurul lucru ce ar putea cauza probleme de conectivitate în exterior este protocolul DNS. Pentru a verifica dacă serverul DHCP a oferit o adresă pentru un server de nume, se va consulta fişierul `/etc/resolv.conf`:

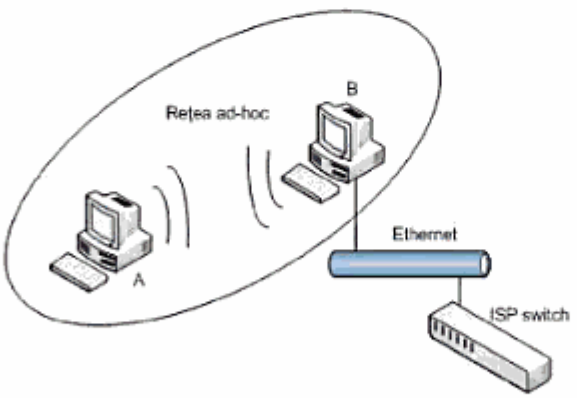
```
waters@myr:~$ cat /etc/resolv.conf

nameserver 82.76.253.115
```

Dacă există un server de nume configurat, problema reţelei părăseşte responsabilitatea administratorul şi trece de partea ISP-ului. Pasul următor pentru rezolvarea problemei este contactarea ISP-ului şi raportarea problemei.

5.3 *Configurarea unei reţele wireless în Linux - configurări avansate*

5.3.1 Partajarea unei conexiuni la Internet într-o reţea *ad hoc*

Una dintre configuraţiile ce se poate întâlni des în funcţionarea reţelelor atât pe fir cât şi fără fir este partajarea unei conexiuni la Internet. Scenariul acesta presupune cel puţin două PC-uri aflate într-o reţea *ad hoc* fără fir sau o reţea *Ethernet* pe fir.



5-11: Partajarea unei conexiuni la Internet

Staţia B dispune şi de o legătură către Internet (deci are 2 interfeţe de reţea montate) şi poate comunica atât cu staţia A, cât şi cu întreaga reţea WWW. Deşi staţia A nu are decât o interfaţa de reţea, trebuie să îi fie asigurată şi acesteia conectivitate în Internet. Există, în general, două soluţii ce se pot adopta în această situaţie:

Material cu drept de autor

- se cumpără un router pentru rețeaua locală și se conectează cele două stații direct la router.
- se configurează stația B ca *gateway/router* pentru stația A.

Se va analiza pe scurt varianta 2, căci varianta 1 presupune mai mult un efort financiar, decât unul de configurare. Rezultatul urmărit poate fi descris astfel:

- stația A trebuie să știe să trimită tot traficul destinat unei stații în Internet, către stația B. Cu alte cuvinte, stația A trebuie să configureze ca *gateway* pe stația B, folosind o rută statică.
- stația B va trebui să trimită pe interfața `eth0` tot traficul primit pe interfața `wlan0` și destinat în Internet. (pentru ca stația A să poată trimite pachete în Internet)
- stația B va trebui să trimită pe interfața `wlan0` tot traficul primit pe interfața `eth0` și destinat stației A. (pentru ca stația A să poată primi pachete din Internet)

Pentru a putea realiza această configurație este nevoie de un *firewall* Linux pe stația B. Ubuntu are deja inclus un *firewall* în linie de comandă numit `iptables`, însa utilizarea sa va fi studiată ulterior în această carte, iar înțelegerea sintaxei `iptables` nu este necesară în acest capitol pentru a oferi o soluție în problema de față.

Pentru a obține comportamentul dorit, se va instala pachetul `firestarter`, un *firewall* extrem de simplist. Comandă pentru instalare este:

```
waters@myr:~$ sudo apt-get install firestarter
```

După instalare, se poate porni programul prin introducerea `firestarter &` în linia de comandă. Acesta va porni intr-un mod „*wizard*" de configurare în care va trebui specificată interfața din rețeaua locală și interfața ce duce spre *gateway*. Restul configurației de *firewall* va fi făcută de către `Firestarter` automat.

Se indică deci interfața spre *gateway*:

și interfața spre rețeaua locală:

A mai rămas doar configurarea *gateway*-ului pe staţia A. Comanda de mai jos indică staţiei A, să trimită toate pachetele destinate în Internet, staţiei B.

```
waters@myr:~$ sudo route add default gw IP_wlan0_B
```

5.3.2 Configurări de securitate în *wireless*

În secţiunea ce urmează vor fi prezentate atât configuraţiile pe client cât şi cele de pe un router *wireless* Linksys.

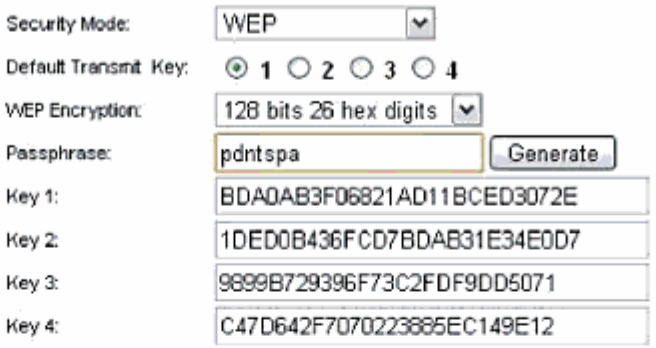
5.3.2.1 Configurarea WEP

5.3.2.1.1 Configurarea routerului wireless

Configurarea routerului se face printr-o interfaţă WEB simplă accesată prin protocolul HTTP. Pentru a putea iniţia însă o conexiune HTTP (nivel 7) la routerul *wireless*, este nevoie de conectivitate de nivel 3 între staţie şi router. Asigurarea legăturii de nivel 1 şi 2 se poate face prin asocierea cu reţeaua implicită *wireless* prezentă pe router sau prin folosirea unui cablu *crossover* pentru conectarea la unul din porturile de LAN. Routerele *wireless* Linksys, au activat în mod implicit un server DHCP ce oferă conectivitatea de nivel 3 prin oferirea unei adrese IP din aceeaşi subreţea cu IP-ul de management al routerului, folosit pentru accesarea interfeţei WEB de configurare. Adresa IP implicită prin care se poate accesa interfaţa *wireless* a routerelor *wireless* Linksys este 192.168.1.1.

5-12: Accesarea routerului prin HTTP

Configurarea WEP, pe router, presupune de fapt alegerea lungimii cheii şi generarea acesteia după o parolă introdusă:



5-13: Configurare WEP pe router wireless

5.3.2.1.2 Configurare clientului wireless

Pe client, tot ce trebuie făcut este adăugarea linie de configurare de mai jos în fişierul `/etc/network/interfaces`.

```
        wireless-key s:7733-7031-7377-3361-6B
```

După repornirea serviciului de reţea, se obţine conectivitate cu suport WEP.

```
root@myr:/home/waters# iwconfig wlan0
wlan0     IEEE 802.11g  ESSID:"test"  Nickname:""
          Mode:Managed  Frequency:2.437 GHz  Access Point: 00:1D:7E:4C:4F:1D
          Bit Rate=54 Mb/s   Tx-Power=27 dBm
          Retry min limit:7   RTS thr:off   Fragment thr=2346 B
          Encryption key:7733-7031-7377-3361-6B
          Power Management:off
          Link Quality=98/100  Signal level=-25 dBm  Noise level=-127 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

Atenţie! *după cum s-a prezentat anterior în subcapitolul de securitate în reţele fără fir, securitatea WEP este foarte uşor de compromis; un utilizator conştient de acest risc nu va trebui să folosească niciodată acest tip de criptare, decât dacă doreşte să îşi spargă propria reţea, în scop didactic.*

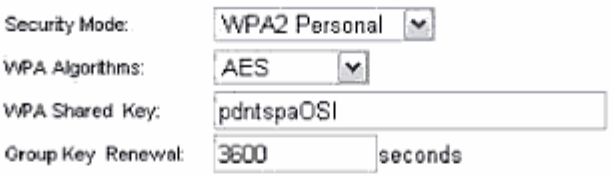
5.3.2.2 Configurarea WPA2 Personal

Standardul de securitate recomandat pentru reţele fără fir de tip infrastructură este 802.11i, cunoscut şi sub numele de WPA2. Configurarea se va face ca şi în cazul WEP, atât pe AP cât şi pe client.

5.3.2.2.1 Configurarea routerului

Configurarea WPA2 se poate face cu protocolul TKIP sau AES, având grijă ca alegerea uneia din opţiuni să se reflecte şi în setările realizate pe client.

Material cu drept de autor



5-14: Configurarea WPA2 pe un router wireless

5.3.2.2.2 Configurarea clientului wireless

Pentru client, se va realiza o configurare persistentă a rețelei pentru suport WPA2. În acest sens va trebui editat fișierul `/etc/network/interfaces` pentru a reflecta schimbarea nivelului de securitate.

> ***Atenție!*** *dacă configurația ce urmează se realizează pe o placă* wireless *Ralink (driver RTxxx), va trebui instalat driverul* `NDISwrapper` *în locul celui* `Serialmonkey`*, prezent în Linux.*

Suportul WPA2 va fi adăugat folosind un utilitar în linie de comandă numit `wpasupplicant`. Se va instala mai întâi pachetul pentru acest utilitar:

```
waters@myr:~$ sudo apt-get install wpasupplicant
```

Pentru a se putea realiza asocierea cu o rețea WPA2, în fișierul de configurare va trebui introdusă cheia partajată WPA2 care a fost setată și pe AP. Problema este că `/etc/network/interfaces` are drept de *read* implicit pentru orice utilizator din sistem. Degeaba s-ar oferi securitate WPA2 rețelei, dacă parola ar fi introdusă în text clar într-un fișier de configurare pe care oricine îl poate consulta. În acest punct intervine `wpasupplicant` prin oferirea unui utilitar cu ajutorul căruia se poate genera un *hash* de 64 caractere, care poate fi ulterior introdus în fișierul `interfaces`. Acest utilitar se numește `wpa-passphrase`, iar pentru a genera *hash*-ul are nevoie la intrare de SSID-ul rețelei și de cheia partajată a rețelei.

```
root@myr:/home/rl# wpa_passphrase YoGi p1d@n3t$s5p^a7
network={
        ssid="YoGi"
        #psk="p1d@n3t^a7"
        psk=a8a9c6966946c520a16d020e7590d1ad35d4de60332a22d7349264007194b0e9
}
```

Alături de cheia partajată, în fișierul de configurare vor mai trebui adăugate următoarele directive:

- **wpa-driver** – specifică driverul folosit. Dacă nu s-a instalat un driver cu ajutorul `NDISwrapper`, se va folosi parametrul **wext** (Linux *wireless* extensions); acesta este driverul generic instalat de Linux.
- **wpa-ssid** – specifică SSID-ul rețelei.
- **wpa-ap-scan** – primește parametrul 1, daca rețeaua are SSID *broadcast* activat, și parametrul 2, în caz contrar.
- **wpa-proto** – versiunea protocolului. Parametrii pot fi WPA2 sau WPA.
- **wpa-pairwise, wpa-group** – aceste două directive primesc același tip de parametru, care specifică procolul de criptare folosit. Valorile pot fi:

1. CCMP pentru AES
2. TKIP pentru TKIP (asigura compatibilitatea WPA2 cu WPA)

- **wpa-key-mgmt** – este folosit pentru a indica metoda de autentificare folosită. Acceptă parametrii:

1. WPA-PSK în cazul autentificării pe bază de cheie partajată
2. WPA-EAP în cazul autentificării pe baza unui server specializat (RADIUS, TACACS+)

Material cu drept de autor

- **wpa-psk** specifică cheia partajată sub formă de *hash* – acceptă ca parametru *hash*-ul generat anterior cu `wpa_passphrase`

Conform specificațiilor de mai sus, fișierul de configurare pentru rețeaua test arată astfel:

```
auto wlan0
iface wlan0 inet dhcp
wpa-driver wext
wpa-ssid YoGi
wpa-ap-scan 1
wpa-proto WPA2
wpa-pairwise CCMP
wpa-group CCMP
wpa-key-mgmt WPA-PSK
wpa-psk a8a9c6966946c520a16d020e7590d1ad35d4de60332a22d7349264007194b0e9
```

5.3.2.3 Compromiterea unui sistem pe bază de cheie WEP

Înainte de a începe compromiterea efectivă, trebuie luate în considerare câteva caracteristici pe care WEP le deține. În primul rând, WEP folosește pentru criptarea datelor un algoritm de criptare simetrică denumit RC4, folosind pentru criptarea și decriptarea datelor, chei pe 40 sau 60 de biți. Cheile sunt cunoscute de AP (*Access Point*) și de toate nodurile cărora li se permite asocierea.

IV (*Initialization Vector*) reprezintă un număr pe 24 de biți, generat aleator de fiecare stație ce dorește să transmită și care împreună cu cheia partajată va genera cheia RC4. Acesta este transmis în clar cu fiecare pachet destinației. Vulnerabilitatea acestui protocol provine tocmai din folosirea acestor vectori de inițializare. Cu o lungime de numai 24 de biți, într-o rețea *wireless* în care există un trafic susținut, WEP până la urmă va folosi aceeași vectori de inițializare pentru diferite pachete. Aceasta rezultă în pachete ce au șiruri generate de către RC4 foarte asemănătoare.

În concluzie, dacă există destul de multe IV-uri capturate, este posibilă aflarea cheii WEP. Modul `managed` sau *ad hoc* al driverului nu suportă însă captura de pachete fără ca clientul să fie asociat cu un AP. Pentru a face posibilă captura trebuie ca placa de rețea să fie setată în modul `Monitor`.

```
waters@myr:~$ sudo ifconfig wlan0 down
waters@myr:~$ sudo iwconfig wlan0 mode Monitor
waters@myr:~$ sudo ifconfig wlan0 up
```

Captura pachetelor și spargerea cheii WEP se vor realiza cu pachetul `aircrack-ng`. Acesta se poate instala folosind utilitarul `apt` în linie de comandă:

```
waters@myr:~$ apt-get install aircrack-ng
```

Pachetul include următoarea suită de utilitare:

- `airodump` – face posibila captura de pachete și implicit de valori IV
- `aircrack` – folosit pentru analiza capturii de IV-uri si găsirea cheii WEP
- `airreplay` – acesta este un *packet injector*. Cu ajutorul său se poate genera trafic la care AP-ul să fie obligat să răspundă cu pachete ce include valori IV.

În scenariul ce urmează se va folosi mai întâi `airodump` pentru a captura pachetele din rețea într-un fișier de pe disc. Asupra acestuia se va aplica `aircrack` pentru a analiza vectorul de IV și a găsi parola.

Odată setat modul `Monitor`, se poate porni captura de pachete cu `airodump`. Ca parametri se vor folosi:

- `-t`: specifică tipul de pachete criptate ce trebuie capturat. Valorile posibile sunt WEP, WPA, WPA2.
- `-b`: specifică banda în care funcționează rețeaua *wireless*

Material cu drept de autor

- -w: specifică numele fişierul în care se va analiza captura
- -c: pentru precizarea canalului care va fi scanat. Dacă nu se precizează canalul scanarea se va face pe toate canalele

Notă: *se presupune că informaţii precum banda sau canalul pe care reţeaua funcţionează au fost aflate anterior printr-o scanare simplă cu utilitarul* `iwlist`.

```
waters@myr:~$ sudo airodump-ng -t WEP -b g -c 11 -w capture  wlan0
[ CH 11 ][ Elapsed: 53 s ][ 2008-08-05 11:57
 BSSID              PWR  RXQ    Beacons  #Data, #/s  CH  MB  ENC  CIPHER AUTH ES
00:1D:7E:4C:4F:1D    0   100       523       50    1   11  54. WEP      WE    1

 BSSID                  STATION            PWR   Rate  Lost  Packets  Probes
 00:1D:7E:4C:4F:1D  00:1D:D9:5D:8F:00        0   54-54     2      235
```

Deşi captura de mai sus a fost lăsată rulând 5 minute, se poate observa că numărul de pachete transmis de staţia cu adresa MAC 00:1D:D9:5D:8F:00 a fost doar 235. Pentru a putea sparge o cheie WEP de 128 biţi cu o probabilitate de 100% va fi nevoie ca indicatorul de date utile(*#Data*) să fie cel puţin 100000 iar numărul de pachete cel puţin la fel de mare. Din păcate, momentan, în reţeaua de mai sus nu are loc foarte mult transfer de date. Deşi s-ar putea aştepta până când activitatea de pe mediu ar fi ceva mai mare, pachetul `aircrack-ng` oferă o metodă mai rapidă.

Folosind utilitarul `airepaly`, se pot trimite pachete ARP *request* la care AP-ul este obligat să răspundă. Din pachetele de ARP *reply* se pot captura IV-urile de care este nevoie. Însă AP-ul nu va răspunde niciodată la un *request* ce vine de la o adresa MAC de nu este asociată reţelei. Din acest motiv, pentru ca acest atac sa funcţioneze, pachetele ARP vor trebui trimise cu un MAC sursă pe care AP-ul îl cunoaşte în tabela sa ARP.

Una din cele mai importante statistici din rezultatul pe care îl oferă comanda `airodump-ng`, este un tabel al staţiilor asociate deja la reţea ce conţine şi adresele MAC ale acestor staţii. Astfel se va folosi drept MAC sursă unul dintre MAC-urile statiilor obţinute anterior.

Sintaxa comenzii `aireplay` ce va fi folosită în acest caz este:

aireplay-ng *<tipul pachetelor injectate> -b <Adresa MAC a AP-ului> -h <Adresa MAC a unui client asociat> <interfaţa de reţea>*

Tipul pachetelor ARP *request* este identificat de numărul 3. Pentru mai mulţi parametrii şi mai multe tipuri de pachete ce pot fi generate, se va consulta pagina `man` a utilitarului. Se poate deci porni injectarea de pachete astfel:

```
waters@myr:~$ sudo aireplay-ng -3 -b 00:1D:7E:4C:4F:1D -h 00:1D:D9:5D:8F:00 wlan0
```

Notă: *în atacul de mai sus, nu contează dacă AP-ul are configurată securitate bazată pe filtrare de adrese MAC, căci pentru pachetele ARP se foloseşte ca adresa sursă, o adresă deja asociată, nefiltrată.*

Acum că s-a generat destul trafic, pentru decriptarea pachetelor de date capturate se va folosi utilitarul `aircrack-ng`, specificându-i numele fişierului în care s-au salvat datele capturate.

```
waters@myr:~$ sudo aircrack-ng  capture.cap
Opening capture.cap
[...]
                        KEY FOUND! [ AA:BB:CC:DD:EE ]
        Probability: 100%
```

Material cu drept de autor

5.4 Wireless în Windows Server 2008

Window Server 2008 vine cu o multitudine de schimbări şi îmbunătăţiri din seria protocoalelor 802.11. Suportul pentru *wireless* este acum nativ, împreună cu o arhitectură de implementare care aduce mari îmbunătăţiri interfeţei de pe partea utilizatorului, oferă posibilitatea stabilirii de politici de grup pentru conexiunile *wireless*, capabilităţi de autoconfigurare. De asemenea, suportă standardul de securitate WPA2 (*Wi-Fi Protected Access 2*). Sistemul încorporat de suport pentru *wireless* se integrează acum cu NAP[1] la autentificarea prin 802.1x, oferă posibilitatea de diagnosticare şi suportă configurarea din linie de comandă.

***Aplicabilitate**. Cu excepţia metodei pentru activarea serviciului wireless, explicată mai jos, majoritatea procedurilor şi utilitarelor descrise în continuare se aplică deopotrivă pentru* Windows Server 2008 *cât şi pentru Windows* Vista.

5.4.1 Activarea serviciului *Wireless* în *Windows Server 2008*

În general, la instalarea sistemului, *Windows Server 2008* va instala automat *driver*-ele pentru majoritatea echipamentelor de reţea. Interfeţele ce nu sunt niciodată configurate automat sunt cele *wireless*. De asemenea, în cele mai multe dintre cazuri, nici instalarea manuală a driverelor pentru aceste componente nu va reuşi. Acest lucru se întâmplă deoarece capabilităţile de *wireless* ale lui *Windows Server 2008* sunt cumulate într-un serviciu *wireless*, inactiv în mod implicit. Odată activat, acest serviciu se va ocupa atât de managementul conexiunilor *wireless* cât şi de centralizarea profilurilor care vor permite conectarea utilizatorilor la reţele *wireless*. Instalarea serviciului e posibilă indiferent de prezenţa unei interfeţe *wireless* în sistem.

Pentru a activa capabilităţile de *wireless* ale lui *Windows Server 2008* este necesară instalarea acestora ca *feature* al sistemului de operare. Pentru aceasta, se execută paşii următori:

- Din meniul `Start > Administrative Tools > Server Manager` sau direct din *Quick Launch* se porneşte *Server Manager* în subcategoria *Features* şi se localizează *Wireless LAN Service* în lista de *feature*-uri.
- Se apasă pe **Next** şi se confirmă instalarea:
- O instalare terminată cu succes va afişa următorul mesaj, la categoria *Results*:

5-15: Instalarea serviciului de *Wireless* terminată cu succes

5.4.2 Configurarea profilurilor *wireless*

O conexiune configurată la o reţea *wireless* este denumită un **profil** *wireless*. Modalităţile prin care aceste profiluri pot fi configurate sunt următoarele:

[1] *Network Access Protection*: http://technet.microsoft.com/en-us/network/bb545879.aspx

Material cu drept de autor

- **Fereastra Connect to a network**, principala metodă accesibilă utilizatorilor individuali pentru a-și configura conexiunea la o rețea *wireless*
- **Politici de grup** (*Group policy*) accesibile administratorilor într-un mediu *Active Directory*[1] pentru a configura centralizat și a distribui configurația altor calculatoare membre ale domeniului.
- **Linie de comandă**, folosind utilitarul `netsh.exe` și comanda `netsh wlan`, ce permite configurarea manuală a rețelelor *wireless*. `Netsh` permite, de asemenea, exportarea profilurilor *wireless* în fișiere xml și importarea lor ulterioară, eventual pe alte sisteme.

5.4.3 Conectarea la o rețea *wireless*

Accesarea interfeței *Connect to a network* poate fi realizată în mai multe moduri:

- Din fereastra *Manage network connections*, accesibilă din `Control Panel > Network and sharing center > Manage network connections`.
- Din meniul de *Connect/Disconnect* al ferestrei *Manage Wireless Connections*, accesibilă la `Control Panel > Network and sharing center > Manage wireless connection`.
- Dacă este prezent, prin clic pe pictograma *Network din System Tray*, urmat de clic pe opțiunea *Connect or disconnect...*
- Doar pe Windows *Vista*, prin accesarea opțiunii *Connect to...* din meniul Start.

5-16: Starea conexiunilor (System Tray > Network)

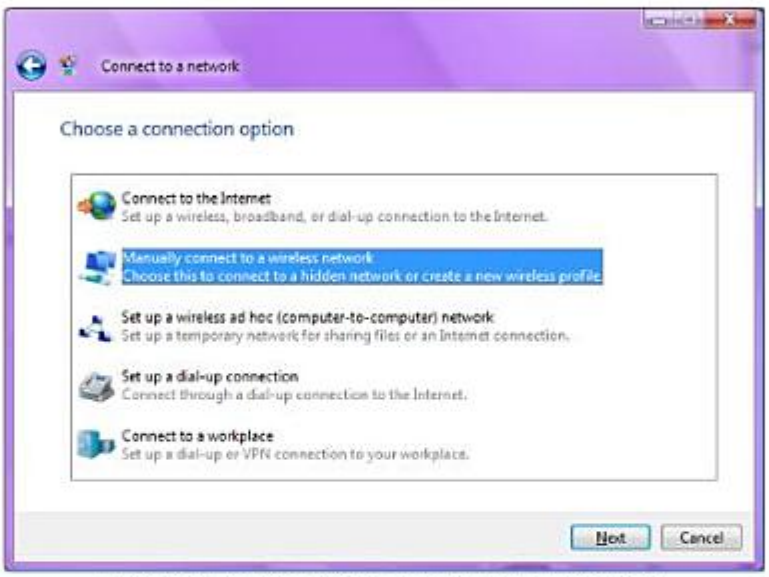
[1] http://www.microsoft.com/windowsserver2008/en/us/active-directory.aspx

Material cu drept de autor

5-17: Fereastra "Connect to a network"

Fereastra *Connect to a network* înlocuieşte vechea fereastră *de Choose a wireless network* din Windows XP SP2. Aceasta suportă acum şi conexiunile VPN (*Virtual Private Network*) precum şi conexiunile de tip *dial-up*, inclusiv *PPPoE* (*Point-to-Point Protocol over Ethernet*). Pentru conectarea la una dintre reţelele din listă, este suficient un dublu-clic pe reţeaua dorită sau selectarea ei şi apăsarea pe *Connect*. Dacă s-a configurat o reţea de *tip non-broadcasting*, aceasta va apărea în listă sub numele de *Unnamed network* iar încercarea de conectare la ea va cere introducerea numelui reţelei[1].

Pentru crearea unei noi conexiuni, din fereastra de mai sus se alege opţiunea *Set up a connection or network.*



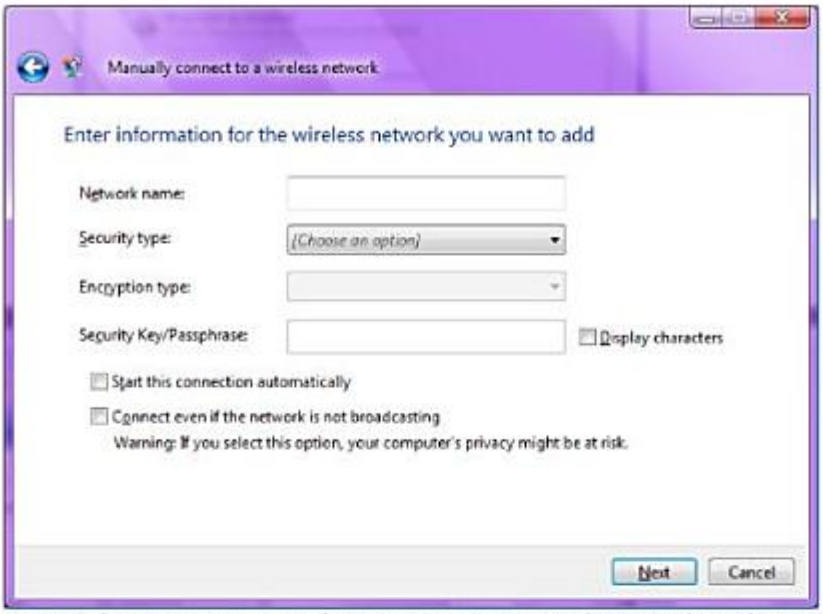
5-18: Alegerea tipului de conexiune ce va fi creată

Pentru conectarea la o reţea *non-broadcasting* la pasul precedent se alege *Manually connect to a wireless network* şi se completează parametrii reţelei:

> ***Atenţie!*** *Dacă opţiunea Manually connect to a wireless network nu este disponibilă, se recomandă verificarea instalării corecte a adaptorului wireless şi identificarea corectă a sa din fereastra Manage network connections.*

[1] Detectarea unei reţele wireless non broadcasting este posibilă deoarece unele AP-uri pot fi configurate astfel încât să trimită cadre de tip *beacon* având câmpul de SSID setat pe valoarea NULL. Mai multe detalii despre cadrele *beacon* la http://en.wikipedia.org/wiki/Beacon_frame.

Material cu drept de autor



5-19: Configurarea parametrilor pentru o reţea *wireless non-broadcasting*

Informaţiile cerute pentru conectare sunt următoarele:

- **Network name**: reprezintă SSID-ul reţelei
- **Security type:** descrie metoda de autentificare în reţea. Sunt permise:
- **No authentication (open)**
- **WEP**
- **WPA:**
  - **Personal**
  - **Enterprise**
- **WPA2:**
  - **Personal**
  - **Enterprise**
- **802.1x:** autentificare IEEE 802.1x cu WEP (*dynamic WEP*)

Autentificarea de tip *shared key* nu este inclusă în listă. Aceasta poate fi configurată ulterior dar Microsoft nu recomandă utilizarea ei deoarece oferă un nivel extrem de scăzut de securitate.

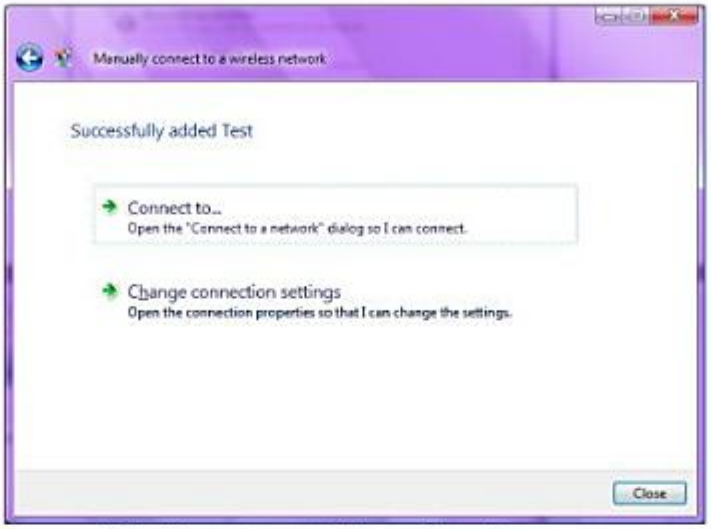
- **Encryption type** reprezintă metoda folosită pentru criptarea cadrelor de date trimise în reţea. Opţiunile disponibile depind de metoda aleasă de autentificare:
  - Pentru **No authentication (open)** se poate selecta **None**
  - Pentru autentificarea **WEP** şi **802.1x** se poate selecta **WEP**
  - Pentru diverse variante **WPA** se poate selecta **TKIP** sau **AES**.

Opţiunile disponibile la criptare depind, de asemenea, şi de capabilităţile interfeţei de reţea *wireless* şi a driverului pe care aceasta îl foloseşte.

- **Security key / Passphrase:** Se introduce cheia **WEP** în cazul securităţii de tip **WEP**, cheia partajata **WPA** sau **WPA2** pentru variantele **Personal** ale acestora, iar pentru variantele **Enterprise** şi **802.1x**, cheia se determină automat la realizarea autentificării.
- **Start this connection automatically:** Windows se va conecta automat la reţea când aceasta este detectată. Altfel, conectarea trebuie facută manual prin fereastra *Connect to a network*.

Material cu drept de autor

- **Connect even if the network is non broadcasting**: Windows va încerca să se conecteze la rețea chiar și când aceasta nu își anunță SSID-ul prin *broadcast*. Se vor trimite cadre de tip *probe request*[1] ce reprezintă un risc de securitate deoarece acestea conțin numele rețelei căutate.



5-20: Noua rețea *wireless* adăugată cu succes

După conectarea cu succes la rețeaua specificată, Windows o va afișa în fereastra *Connect to a network*. Din ecranul de confirmare a creării noii rețele se poate alege opțiunea *Change connection settings*, ce va permite modificarea parametrilor de securitate și de conectare automată. Tot aici, la secțiunea de securitate, se poate alege opțiunea *Shared* (nedisponibilă la crerea rețelei), deși nu este recomandată datorită securității scăzute.

În funcție de tipul de securitate ales, se configurează fie o cheie de autentificare fie o metodă de autentificare în rețea. În cazul din urmă, deci la configurarea unei autentificări de tip *WPA-Enterprise*, *WPA2-Enterprise* sau *802.1x*, sunt necesare și următoarele configurări ulterioare:

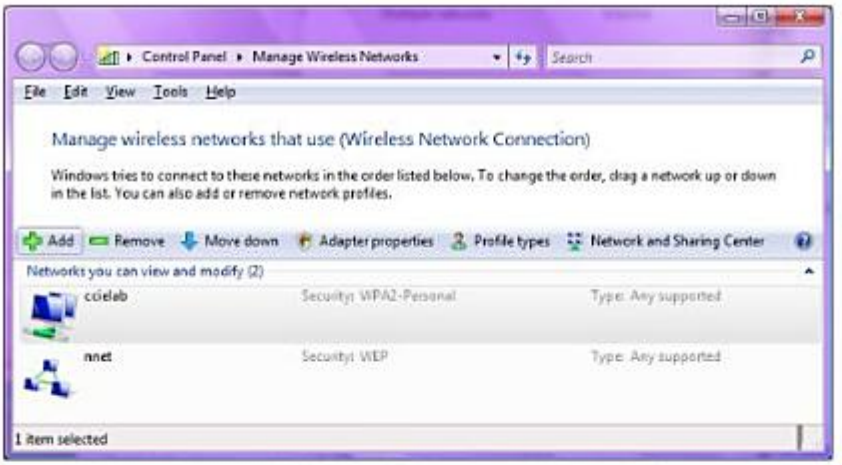
- **Choose a network authentication method:** Se alege o metodă EAP (*Extensible Authentication Protocol*) și se apasă *Settings* pentru a configura tipul de EAP ales.
- **Cache user information for subsequent connections to this network**: Opțiune care specifică faptul că, atunci când utilizatorul se deconectează din sistem, datele folosite pentru autentificare îi vor fi sau nu șterse din *registry*. În cazul în care sunt șterse, datele vor fi cerute din nou la fiecare autentificare în rețea.

5.4.4 Managementul conexiunilor *wireless*

După crearea și/sau detectarea cu succes a rețelelor *wireless*, managementul acestora poate fi realizat dintr-o interfață specializată pusă la dispoziție de *Windows Server 2008* și accesibilă prin `Control Panel > Network and sharing center > Manage` ***wireless*** `networks`.

[1] Mai multe detalii despre tipurile de cadre în *wireless* la adresa: http://www.wi-fiplanet.com/tutorials/article.php/1447501

Material cu drept de autor



5-21: Fereastra *Manage wireless networks*

În fereastra *Manage wireless networks* pot fi vizualizați parametrii cu care au fost configurate rețelele *wireless*, li se pot modifica proprietățile și, bineînțeles, pot fi șterse sau adăugate noi profiluri *wireless*. De asemenea, tot de aici se pot rearanja în ordinea preferințelor profilurile *wireless* configurate, ordine folosită de către Windows atunci când sunt detectate una sau mai multe rețele *wireless*.

Deși managementul rețelelor *wireless* poate fi realizat prin *interfața Manage wireless connections*, conexiunea *wireless* curentă poate fi configurată și prin interfața *Manage network connections*, cu diferența că aceasta va afișa toate tipurile de conexiuni prezente în sistem, atât cele detectate automat (cum ar fi interfețele *Ethernet*) cât și cele configurate manual (spre exemplu, conexiuni *Bluetooth* sau PPPoE), exemplificate în imaginea 5-22:

5-22: Fereastra *Manage network connections*

5.4.5 Conexiuni *wireless* ad hoc

O rețea *ad hoc*[1] mai este denumită și rețea *computer-to-computer*, deoarece calculatoarele se conectează direct, fără a mai folosi dispozitive intermediare ca huburi, switchuri sau routere. Avantajul unei rețele *wireless ad hoc* este că poate fi instalată practic

[1] De fapt, o rețea ad hoc presupune o rețea *wireless* și nu are sens în alt context.

Material cu drept de autor

oriunde cu destul de multă uşurinţă, poate fi folosită în orice scop (partajarea fişierelor, jocuri în reţea) şi permite chiar şi partajarea unei conexiuni la Internet.

Pentru a crea o reţea ad hoc în *Windows Server 2008*, se urmează secvenţa următoare de paşi:

1. Se deschide interfaţa *Connect to a network* (prezentată în 5.4.3)
2. Se alege *Set up a connection or network*
3. Din lista tipurilor de conexiuni disponibile, se alege *Set up an ad hoc (computer-to-computer) network* şi se apasă *Next*.

5-23: Configurarea unei reţele ad hoc

4. În fereastra următoare se completează numele (SSID-ul) reţelei, se alege tipul de securitate implementată (nu orice placă *wireless* suportă WPA2 în mod *ad hoc*) şi se introduce *passphrase*-ul reţelei.

5-24: Reţeaua ad hoc creată cu succes

5. Windows anunţă faptul că reţeaua a fost creată cu succes. Noua reţea apare acum în interfaţa *Connect to a network* iar sistemul se conectează automat la ea.

5-25: Reţeaua ad hoc nou creată, în interfaţa *Connect to a network*

Material cu drept de autor

Atenţie! *După cum s-a menţionat mai sus, după crearea unei reţele wireless sistemul se conectează automat la ea, ceea ce implică faptul că orice altă conexiune wireless ce era activă în acel moment va fi deconectată.*

Dacă nu se bifează opţiunea *Save this network* (figura 5-23) profilul reţelei nou create va fi şters automat în momentul în care ultimul client se deconectează de la ea sau când cel care a creat reţeaua se deconectează sau iese din raza celorlaltor clienţi.[1]

5.5 *Administrarea în linie de comandă şi* PowerShell

5.5.1 Managementul serviciului *wireless* prin netsh wlan

`netsh` este un utilitar disponibil în versiunile de *Windows 2000*, XP, 2003 şi 2008/*Vista*. `Netsh.exe` poate fi folosit pentru configurarea interfeţelor de reţea, a protocoalelor de rutare, poate aplica filtre, poate modifica rute şi poate seta o multitudine de parametri ai interfeţelor de reţea.

Parametrul `wlan` reprezintă un „context" al comenzii `netsh` şi este folosit împreună cu aceasta pentru a efectua modificări asupra configuraţiilor reţelelor *wireless*.

Pentru a folosi comanda, se introduce `netsh wlan` fie în promptul de comandă `cmd.exe`, fie în *PowerShell*.

5.5.1.1 Comenzi de tip "show"

Pentru o listă completă a parametrilor ce pot fi folosiţi împreună cu forma `show` a `netsh` wlan, se introduce una dintre comenzile:

```
PS C:\Users\Administrator> netsh wlan show
PS C:\Users\Administrator> netsh wlan show ?
```

O serie importantă de informaţii sunt oferite de comanda `netsh wlan show drivers`; comanda afişează capabilităţile driverului interfeţei *wireless* instalate în sistem. Se pot identifica tipurile de standarde suportate, protocoalele de securitate ce pot fi folosite, precum şi informaţii despre driverul propriu-zis. Opţional, se poate adăuga comenzii parametrul `interface`, cu sintaxa `interface="`*Wireless* `Network Connection"`, prin care se specifică o anumită interfaţă *wireless*:

```
PS C:\Users\Administrator> netsh wlan show drivers

Interface name: Wireless Network Connection

    Driver                    : Dell Wireless 1390 WLAN Mini-Card
    Vendor                    : Broadcom
    Provider                  : Broadcom
    Date                      : 10/12/2007
    Version                   : 4.170.25.17
[...]
    Type                      : Native Wi-Fi Driver
    Radio types supported     : 802.11g 802.11b
    FIPS 140-2 mode supported : No
    Authentication and cipher supported in infrastructure mode:
                                Open            None
                                Open            WEP
                                Shared          None
                                Shared          WEP
                                WPA2-Enterprise TKIP
                                WPA2-Personal   TKIP
                                WPA2-Enterprise CCMP
                                WPA2-Personal   CCMP
                                Unknown         TKIP
```

[1] Puteţi citi un articol interesant despre pericolele reţelelor ad hoc la http://www.windowsecurity.com/whitepapers/Dangers-Ad-Hoc-*Wireless*-Networking.html

Material cu drept de autor

```
                                    Unknown         CCMP
                                    WPA-Enterprise  TKIP
                                    WPA-Personal    TKIP
                                    WPA-Enterprise  CCMP
                                    WPA-Personal    CCMP
        Authentication and cipher supported in ad-hoc mode:
                                    WPA2-Personal   CCMP
                                    Open            None
                                    Open            WEP
    [...]
```

5-26: Capabilităţile interfeţei *wireless*

Pentru afişarea interfeţelor *wireless* din sistem în contextul reţelelor la care acestea sunt conectate, se foloseşte comanda `netsh wlan show interfaces`:

```
PS C:\Users\Administrator> netsh wlan show interfaces

There is 1 interface on the system:

    Name                   : Wireless Network Connection
    Description            : Dell Wireless 1390 WLAN Mini-Card #3
    GUID                   : 6fe1ef65-14ac-4a72-bf4b-52a821535ace
    Physical Address       : 00:19:7e:11:91:64
    State                  : connected
    SSID                   : DLINK_WIRELESS
    BSSID                  : 00:19:5b:22:31:a4
    Network Type           : Infrastructure
    Radio Type             : 802.11g
    Authentication         : Open
    Cipher                 : None
    Connection Mode        : Auto Connect
    Channel                : 6
    Receive Rate (Mbps)    : 54
    Transmit Rate (Mbps)   : 54
    Signal                 : 60%
    Profile                : DLINK_WIRELESS
```

Pentru afişarea listei reţelelor *wireless* configurate, echivalentul listei obţinute prin interfaţa *Connect to a network*, se poate folosi comanda `netsh wlan show profiles`:

```
PS C:\Users\Administrator> netsh wlan show profiles

Profiles on interface Wireless Network Connection:

Group Policy Profiles (read only)
---------------------------------
    <None>

User Profiles
-------------
    All User Profile     : DLINK_WIRELESS
    All User Profile     : ccielab
    All User Profile     : nnet
```

Afişarea unei liste a reţelelor *wireless* detectate se face prin comanda `netsh wlan show networks`:

```
PS C:\Users\Administrator> netsh wlan show networks

Interface Name : Wireless Network Connection
There are 5 networks currently visible.

SSID 1 : YoGi
    Network type            : Infrastructure
    Authentication          : WPA-Personal
    Encryption              : CCMP

SSID 2 : ccielab
    Network type            : Infrastructure
    Authentication          : WPA2-Personal
    Encryption              : CCMP

SSID 3 : Bee
    Network type            : Infrastructure
    Authentication          : Open
    Encryption              : WEP
[...]
```

Material cu drept de autor

Comanda `netsh wlan show` mai suportă parametri ca `autoconfig`, `blockednetworks`, `filters`, `settings`, `tracing`. Pentru a afişa toate informaţiile disponibile pentru toate interfeţele *wireless* şi toate reţelele, detectate sau configurate, se foloseşte parametrul `all`:

```
netsh wlan show all
```

5.5.1.2 Salvarea, încărcarea şi ştergerea profilurilor

Netsh oferă posibilitatea exportării în fişiere xml a configuraţiilor reţelelor *wireless*. Pentru a exporta o astfel de configuraţie se foloseşte sintaxa comenzii `netsh wlan export profile folder=` cu următorii parametri:

- `folder` = *cale şi nume fişier*; poate fi o cale absolută sau relativă
- `name` = opţional, *numele profilului* de exportat
- `interface` = opţional, *numele interfeţei* pe care profilul este configurat

Dacă se include parametrul `name`, atunci se va exporta doar un singur profil. Dacă se specifică parametrul `interface`, se salvează toate profilurile configurate pe acea interfaţă. Lipsa parametrului `interface` are ca efect exportarea tuturor profilurilor din sistem.

Exemplu de utilizare, cu salvarea profilurilor în directorul curent (.):

```
PS C:\Users\Administrator> netsh wlan export profile folder=.

Interface profile "DLINK_WIRELESS" is saved in file ".\Wireless Network Connection-
DLINK_WIRELESS.xml" successfully.

Interface profile "ccielab" is saved in file ".\Wireless Network Connection-ccielab.xml"
successfully.

Interface profile "nnet" is saved in file ".\Wireless Network Connection-nnet.xml"
successfully.
```

Încărcarea configuraţiilor din fişierele xml exportate se face prin utilizarea parametrului `add profile`, ca în următorul exemplu:

```
netsh wlan add profile filename= C:\"my_wlan_profile.xml" user=all interface= "Wireless
Network Adapter"
```

Opţional, mai pot fi folosiţi doi parametri suplimentari: parametrul `user`, care dacă rămâne nespecificat, importă setările automat pentru utilizatorul curent sau poate primi una dintre valorile `all` sau `current`, iar parametrul `interface` pentru a specfica o anumită intrefaţă *wireless* pentru care profilul să fie aplicat.

La specificarea intefeţei, se pot folosi simbolurile `?` şi `*` (*wildcard*) pentru aproximarea denumirilor.

Exportarea modificărilor efectuate se poate realiza şi prin parametrul `dump`, care va exporta un script ce poate fi executat pentru a reface o configuraţie intermediară, spre exemplu:

```
netsh wlan dump > c:\wlandump.txt
```

5.5.1.3 Conectarea şi deconectarea

O alternativă la interfaţa *Connect to a network* o reprezintă comenzile de conectare şi deconectare de la reţele *wireless* utilizabile prin intermediul lui `netsh wlan`. Atât conectarea cât şi deconectarea se realizează prin intermediul numelui profilului configurat pentru acea reţea.

Comenzile se folosesc în modul următor:

```
connect ssid=linksys name=linksys_profile interface="Wireless Network Adapter"
```

Comanda anterioară realizează conectarea la o reţea iar următoarea, deconectarea:

```
disconnect interface="Wireless Network Adapter"
```

Material cu drept de autor

Dacă în sistem este instalată doar o singură interfață *wireless*, parametrul `interface` poate să lipsească. În cazul în care se încearcă conectarea la o rețea folosindu-se o interfață care este deja conectată la o altă rețea, se va realiza deconectarea de la rețeaua anterioară și conectarea la cea dată ca parametru. Conectarea la aceeași rețea la care interfața este deja conectată va returna un mesaj care anunță conectarea cu succes, dar starea conexiunii nu va fi modificată în niciun fel.

5.5.1.4 Filtre și profiluri

`netsh` oferă posibilitatea de a aplica filtre pe baza SSID-urilor rețelelor *wireless*. Cu alte cuvinte, accesul (conectarea) la anumite rețele poate fi blocat sau permis în mod explicit. Pentru managementul filtrelor, se pot folosi parametrii `add filter` și `delete filter` ca în exemplele următoare:

```
netsh wlan add filter permission=block ssid=linksys networktype=infrastructure
```

Comanda de mai sus realizează adăugarea unui filtru care interzice conectarea la rețele de tip infrastructură cu numele *„linksys"*. Permisiunile posibile sunt de tipul `allow`, `block` sau `denyall`. De asemenea, tipul rețelei dat prin parametrul `networktype` poate fi `infrastructure` sau `adhoc`.

De asemenea, se folosește o comandă similară pentru ștergerea unui filtru:

```
netsh wlan delete filter permission=block ssid=linksys networktype=infrastructure
```

Atât în cazul adăugării cât și în cazul ștergerii de filtre, utilizarea parametrului `ssid` este necesară doar pentru valorile de `allow` sau `block` ale parametrului `permission`. Dacă se creează sau se șterge un filtru cu permisiunea `denyall`, parametrul `ssid` nu trebuie inclus.

Definirea profilurilor *wireless* este, de asemenea, posibilă prin intermediul lui `netsh wlan`. Pentru a adăuga sau șterge profiluri se folosesc parametrii `add profile` sau `delete profile`, ca în exemplele următoare, pentru adăugare, respectiv, ștergere de profiluri:

```
netsh wlan add profile filename="my_wlan_profile.xml" interface="Wireless Network
Adapter" user=all

netsh wlan delete profile name="linksys_profile interface="Wireless Network Adapter"
user=all
```

Adăugarea profilurilor necesită specificarea fișierului din care să fie încărcate setările. Din acest fișier se încarcă și numele propriu-zis al profilului, ce este folosit mai apoi pentru comanda de ștergere. Trebuie ținut cont de faptul că setările stocate în fișiere de tip profil nu sunt încărcate decât la comenzi de tip `add profile`, configurația curentă fiind stocată în sistemul de operare, astfel că modificarea unui fișier profil fără încărcarea lui nu va avea niciun efect (cu alte cuvinte, nu sunt considerate fișiere de configurare).

Parametrul `interface` este opțional în ambele cazuri și cere folosirea unui nume de interfață conform modului în care ele sunt afișate prin comanda `netsh wlan show interfaces`. În momentul folosirii lui, adăugarea sau ștergerea profilurilor va avea efect doar pe interfața respectivă. De asemenea, specificarea parametrului `user` este opțională și va efectua modificările pentru utilizatorul respectiv. Altfel, comenzile de creare sau ștergere profiluri se vor aplica doar utilizatorului curent.

Pentru ștergerea profilurilor, omiterea parametrului `interface` va avea ca efect ștergerea profilului respectiv de pe toate interfețele active.

5.5.1.5 Comenzi de tip "set"

Comanda `set autoconfig` are rolul de a activa sau de a dezactiva serviciul de autoconfigurare pe o anumită interfață. Dacă serviciul de autoconfigurare este activ, *Windows*

Material cu drept de autor

Server 2008 se va conecta automat la rețelele *wireless* prin interfața corespunzătoare. Implicit, acest serviciu este activ.

Un exemplu de utilzare este următorul:

```
netsh wlan set autoconfig enabled=yes inteface="Wireless Network Adapter"
```

Parametrul `enabled` poate accepta valoarea `yes` sau `no`, iar menționarea interfeței este obligatorie.

Comanda `set profileorder` oferă posibilitatea de a atribui priorități profilurilor configurate pentru a defini ordinea în care se preferă conectarea prin acestea.

Exemplul următor definește prioritatea 2 pentru un anumit profil de pe o anumită interfață:

```
netsh wlan set profileorder name=linksys_profile interface="Wireless Network Adapter"
priority=2
```

O valoare mai mică reprezintă o prioritate mai buna, iar dacă valoarea parametrului `priority` este setată pe 1 sau 0, profilul declarat va trece automat pe prima poziție, indiferent dacă mai exista un altul configurat cu aceeasi prioritate.

Comanda `set tracing` permite activarea sau dezactivarea modului în care serviciul *wireless* ține evidența (în jurnalele de sistem) a evenimentelor. În mod implicit, modul *tracing* este dezactivat.

Pentru activarea *tracing*-ului se folosește comanda:

```
netsh wlan set tracing mode=yes
```

Activarea sau dezactivarea modului *tracing* se face prin specificarea parametrului `mode` împreună cu valorile `yes` sau `no`. Deoarece comportamentul implicit al *tracing*-ului este de a se dezactiva în momentul restartării sistemului, pentru menținerea sa activă din momentul autentificării utilizatorilor este necesară includerea modului `persistent`.

5.5.2 Managementul serviciului *wireless* prin *PowerShell*

Pentru a genera o listă sub formă de tabel a tuturor interfețelor de rețea (inclusiv cele virtuale) folosind *PowerShell*, se utilizeaza comanda `Get-WmiObject` în modul următor:

```
PS C:\Users> Get-WmiObject win32_networkadapter | Format-Table

ServiceName      MACAddress        AdapterType      DeviceID        Name
-----------      ----------        -----------      --------        ----
[...]
bcm4sbxp         00:13:D4:9E:5... Ethernet 802.3   6               Broadcom 440x...
BCM43XX          00:19:7E:11:9... Ethernet 802.3   7               Dell Wireless...
[...]
BTWDNDIS         00:19:7D:E1:A... Ethernet 802.3   18              Bluetooth LAN...
[...]
```

5-27: Interfețele de rețea listate prin WMI

Din lista obținută mai sus este important de identificat numele interfeței *wireless* împreună cu identificatorul său (câmpul `DeviceID`) pentru a o putea selecta cu ușurință în continuare:

```
PS C:\Users> Get-WmiObject win32_networkadapter | where {$_.DeviceId -eq 7}


ServiceName      : BCM43XX
MACAddress       : 00:19:7E:11:91:64
AdapterType      : Ethernet 802.3
DeviceID         : 7
Name             : Dell Wireless 1390 WLAN Mini-Card #3
NetworkAddresses :
Speed            : 11000000
```

Material cu drept de autor

Rezultatul lui `Get-WmiObject win32_networkadapter` a fost trimis prin operatorul *pipe* (|) instrucțiunii `where`, care selectează doar intrările ce corespund condiției din paranteză (câmpul `DeviceId` al obiectului curent trebuie să fie 7). Pentru a efectua în continuare operații asupra interfeței selectate, ca obiect, este mai uşor să se păstreze o referință asupra obiectului returnat într-o variabilă:

```
$my_wireless = Get-WmiObject win32_networkadapter | where {$_.DeviceId -eq 5}
```

După ce s-a obținut referința la obiectul creat, acestuia i se pot seta diferite proprietăți sau i se pot apela metodele. Pentru o listă completă a proprietăților şi metodelor disponibile se poate trimite obiectul nou creat comenzii `Get-Member`:

```
PS C:\Users\Administrator> $my_wireless | Get-Member
```

Comanda de mai sus returnează aproximativ 60 de metode şi proprietăți ale interfeței de rețea. Una dintre utilizările simple ale lor reprezintă apelarea metodelor `$my_wireless.Enable()` sau `$my_wireless.Disable()`.

Material cu drept de autor

Întrebări

1. Cel mai popular standard WLAN, în momentul de față, este:

- *802.11a*
- *802.11b*
- *802.11g*
- *802.11n*

2. Dezavantajul undelor din banda de 2,4 GHz față de cele din banda de 5 GHz este:

- *penetrează mai greu materialele*
- *sunt foarte nocive pentru organismele vii*
- *costul de producție ale echipamentelor este mai mare*
- *interferențele sunt mai mari*

3. Câte rețele 802.11b (11 Mbps) pot funcționa, fără interferențe, în aceeași rază de acțiune în banda ISM de 2,4 GHz ?

- *1*
- *2*
- *3*
- *4*

4. În ce mod de funcționare trebuie configurată placa de rețea cu ajutorul utilitarului `iwconfig` pentru a se putea realiza capturi de trafic:

- *Monitor*
- *Managed*
- *Ad-hoc*
- *Promiscuous*

5. Care dintre următoarele metode oferă cel mai mic nivel de securitate:

- *Filtrarea pe bază de MAC*
- *Eliminarea SSID broadcast*
- Securitatea WEP
- Securitatea WPA2

6. Care din următoarele utilitare pot fi folosite pentru configurarea temporară a parametrilor de nivel 2 ai unei rețele *wireless*.

- Ifconfig
- iproute2
- iwconfig
- route

7. Tehnica folosită în protocolul wireless pentru acces la mediu este:

- *CSMA/CD*
- *CSMA/CA*
- *Wireless CSMD*
- *CSMW*

Material cu drept de autor

8. Care din următoarele standarde este un standard de Wireless MAN ?

- *802.11n*
- *802.13z*
- *802.3ab*
- *802.13e*

9. Care din următoarele dispozitive nu poate asocia clienți la o rețea infrastructură?

- *Access point*
- *Router wireless*
- *Bridge*
- *Controller wireless*

10. Undele wireless sunt unde de tip:

- *Microunde*
- *Unde radio de înaltă frecvență*
- *Ultraviolete*
- *Unde audio*

Material cu drept de autor

6 Securitate şi monitorizare

Ce se învaţă din acest capitol?

- Funcţionarea protocolului SSH
- Configurarea de bază şi avansată a protocolului SSH
- Funcţionarea unui *firewall*
- Configurarea de bază şi avansată de filtrare de pachete
- Configurarea Wireshark
- Configurare `snort`
- Configurare Windows Firewall
- Monitorizare în Windows

Cine este...

Bruce Schneier *este un specialist în securitate. Schneier este autorul "Applied Cryptography" şi "Practical Cryptography". Schneier a proiectat sau contribuit la proiectarea mai multor algoritmi de criptare precum Blowfish, Twofish, Helix. În acest moment, Schneier este CSTO la BT Counterpane, companie pe care a înfiinţat-o.*

Daniel J. Bernstein *este matematician şi programator. Actualmente este profesor la Universitatea din Illinois. Bernstein este autorul qmail, publicfile şi djbdns. Bernstein a publicat un număr important de articole în matematică. Bernstein este autorul bibliotecii matematice djbfft folosită pentru calculul FFT.*

Theo de Raadt *este fondatorul şi conducătorul proiectelor OpenBSD şi OpenSSH. De Raadt a fondat OpenBSD după ce a părăsit proiectul NetBSD. Proiectelor OpenBSD şi derivate sunt cunoscute pentru preocuparea pentru securitate. O personalitate abrazivă, Theo de Raadt a avut conflicte cu alţi membri din comunitatea open-source, dar îi sunt recunocute meritele în promovarea driverelor free software şi a modelului deschis de dezvoltare.*

6.1 Secure Shell (SSH)

SSH (*Secure Shell*) este un protocol utilizat pentru accesul la distanţă şi pentru executarea comenzilor pe o maşina de la distanţă. A fost conceput pentru a înlocui `rlogin`, `rsh` şi `telnet` şi pentru a asigura comunicaţie criptată între două staţii ce comunică într-o reţea nesigură, aşa cum este Internetul. Prin canalul oferit pot fi redirectate şi conexiuni X11 şi porturi arbitrare TCP/IP. SSH utilizează conexiuni TCP, componenta server ascultând pe portul 22.

Aşa cum s-a menţionat şi mai sus, SSH oferă o comunicaţie criptată între două staţii, criptarea oferind datelor confidenţialitate şi integritate. SSH utilizează criptografia cu chei publice pentru autentificarea staţiilor ce doresc să se conecteze şi să execute comenzi la distanţă, conectarea realizându-se pe baza unui nume de utilizator şi a unei parole. Sunt suportate următoarele metode de criptare: IDEA, DES, 3DES, ARCFOUR, BLOWFISH şi TSS[1], implicit folosindu-se IDEA.

[1] Capitolul de Securitate din Computer Networks, ediţia a patra, Tanenbaum.

Material cu drept de autor

6-1: Conexiune SSH criptată

În prezent, există două versiuni ale acestui protocol. Prima versiune, SSH-1, a fost lansată în 1995, câştigând rapid popularitatea utilizatorilor. În 1996, apare SSH-2, o rescriere integrală a primei versiuni, incompatibilă cu prima versiune, aducând îmbunătăţiri substanţiale în procesul de schimbare a cheilor şi în asigurarea integrităţii datelor. Principalele diferenţe dintre cele două versiuni sunt listate în tabelul de mai jos:

Caracteristici	SSH v1	SSH v2
Structură	O singură componentă care se ocupă de transport, autentificare şi sesiune	Componente separate pentru transport, autentificare şi sesiune
Suport pentru certificate	Nu	Da
Modificarea periodică a cheilor de sesiune	Nu	Da
Verificarea integrităţii mesajelor	Foloseşte verificarea CRC-32 care poate fi atacată cu atacul pe bază de inserţie[1]	Foloseşte un algoritm de criptare/decriptare puternic

6.1.1 Protocolul SSH

6.1.1.1 Instalare

Cea mai populară implementare a protocolului SSH o reprezintă pachetul `OpenSSH`, un proiect *open source*, lansat şi creat de cei de la OpenBSD. În prezent, cea mai nouă versiune de `OpenSSH` este 4.7p1, lansată pe 4 septembrie 2007.

Pachetul `ssh` este compus dintr-un server (`sshd`), un client (`ssh` – disponibil implicit pe majoritatea distribuţiilor) şi un set de utilitare, pentru gestiunea cheilor. În general `sshd` este pornit de scripturile de iniţializare ale sistemului şi rulează permanent în *background*.

Instalarea serverului SSH include instalarea unor module adiţionale cum ar fi:

- `sshd` – componenta server
- `ssh-keygen` – utilitar folosit pentru generarea cheilor
- `ssh-keyscan` – utilitar folosit pentru administrarea cheilor publice
- `scp` – utilitar pentru copierea sigură de fişiere
- `ssh-agent` – componenta folosită pentru salvarea cheilor private etc.

```
root@myr:~#  apt-get install ssh
Reading package lists... Done
Building dependency tree... Done
The following extra packages will be installed:
  openssh-server
Suggested packages:
```

[1] https://honor.trusecure.com/pipermail/firewall-wizards/1998-June/002845.html

Material cu drept de autor

```
  rssh
The following NEW packages will be installed:
  openssh-server ssh
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 206kB of archives.
After unpacking 586kB of additional disk space will be used.
Do you want to continue [Y/n]? y
[...]
Setting up openssh-server (4.2p1-7ubuntu3) ...
Creating SSH2 RSA key; this may take some time ...
Creating SSH2 DSA key; this may take some time ...
 * Restarting OpenBSD Secure Shell server...                    [ ok ]

Setting up ssh (4.2p1-7ubuntu3) ...
```

Odată instalat, serverul SSH este pornit automat. Pentru pornirea sau repornirea sa se poate folosi scriptul `/etc/init.d/ssh`.

```
root@myr:~# /etc/init.d/ssh -h
Usage: /etc/init.d/ssh {start|stop|reload|force-reload|restart}
```

6.1.1.2 Funcţionarea protocolului SSH

6.1.1.2.1 Criptografia cu chei publice

Criptografia reprezintă procesul de transformare a unui text clar într-un text cifrat. Ea stă la baza multor servicii şi mecanisme de securitate folosite în Internet, folosind metode matematice pentru transformarea datelor, cu intenţia de a ascunde conţinutul lor sau de a le proteja împotriva modificării. Securizarea comunicaţiilor în Internet poate fi comparată cu semnarea unei scrisori şi trimiterea acesteia într-un plic sigilat. Semnătura dă autenticitatea scrisorii, iar plicul sigilat îi conferă acesteia confidenţialitatea necesară.

Electronic, confidenţialitatea este asigurată prin criptarea mesajului cu o cheie secretă folosind un algoritm asociat. Versiunea criptată a mesajului poate fi citită de destinatar numai dacă acesta posedă cheia secretă şi algoritmul de decriptare.

Există două tipuri de sisteme criptografice:

- simetrice (cu chei secrete), care folosesc aceeaşi cheie atât la criptarea cât şi la decriptarea mesajelor.
- asimetrice (cu chei publice), care folosesc chei distincte de criptare şi decriptare. Una din chei este ţinută secretă şi este cunoscută doar de proprietarul ei. A doua cheie (perechea ei) este făcută publică, deoarece este imposibilă deducerea unei chei din cealaltă.

Una dintre metodele de autentificare folosită de protocolul SSH este bazată pe algoritmul RSA (*Rivert-Shamir-Adleman*). Publicat încă din 1977, RSA este un algoritm folosit pentru criptografia cu chei publice.

Criptografia cu chei publice funcţionează în modul următor: o persoană care doreşte să primească mesaje secrete deţine două chei, una publică şi una privată. Cheia publică poate fi afişată pe pagina Web personală sau făcută publică printr-un alt mijloc, aceasta putând fi văzută de către oricine. Cheia privată, în schimb, va fi ţinută secretă pe staţia locală. Dacă în aceste condiţii cineva va dori sa trimită mesaje secrete acestei persoane, va lua cheia publică afişată pe pagina Web personală şi va cripta mesajul. Când mesajul va ajunge la destinaţie, persoana ce deţine cheia privată (perechea cheii publice care a fost utilizată în criptarea mesajului trimis) va decripta mesajul cu ajutorul acesteia. În absenţa cheii private mesajul nu va putea fi decriptat, astfel încât numai destinatarul lui de drept îl va putea citi.

În procesul de instalare a serverului, va fi creată implicit o pereche de chei (publică şi privată), fiecare cheie fiind stocată într-unul dintre fişierele: `/etc/ssh/ssh_host_rsa_key` şi `/etc/ssh/ssh_host_rsa_key.pub`. În fişierul `ssh_host_rsa_key.pub` este salvată cheia publică, iar în fişierul `ssh_host_rsa_key` cheia privată.

Material cu drept de autor

6.1.1.2.2 Stabilirea conexiunii

Fiecare staţie (client) are o cheie privată RSA, *host key* (în mod normal pe 1024 de biţi). În plus, atunci când este pornit, *daemon*-ul ssh (`sshd`) generează automat o a doua cheie, *server key* (pe 768 de biţi). Aceasta este regenerată din oră în oră dacă a fost folosită şi nu este păstrată niciodată pe disc. De fiecare dată când un client iniţiază o conexiune, *daemon*-ul îi trimite *host key* şi *server key* (care este publică). Clientul compară *host key* primit cu cea din baza lui de date, pentru a verifica dacă nu s-a schimbat, apoi generează un număr aleator pe 256 de biţi. Clientul criptează acest număr folosind întâi *host key* şi apoi *server key* şi trimite numărul criptat la server. În continuare ambele părţi vor folosi acest număr aleator ca o cheie de criptare.

6.1.2 Configuraţii de bază SSH

6.1.2.1 Utilizarea serviciului ssh

Comanda de conectare la un server SSH are doi parametrii importanţi: numele utilizatorului şi adresa (numele) serverului destinaţie. Serverul de SSH la care se va face conectarea: `securessh.pub.ro` (în exemplul de mai jos), acest nume este un nume public, pe care serviciul de DNS îl va traduce într-o adresă IP. Dacă se doreşte conectarea la un server dar nu se cunoaşte numele său DNS, se poate introduce direct IP-ul serverului. Când un client iniţiază o conexiune `ssh` pe un server, trebuie să se conecteze pe un utilizator existent pe acel server pentru a putea avea acces la interpretorul de comenzi. Parametrul `bogdand` specifică utilizatorul în contul căruia se va intra la stabilirea conexiunii.

```
waters@myr:/$ ssh bogdand@securessh.pub.ro
Password:
Last login: Wed Sep 19 14:37:29 2007 from 86.121.138.243
Welcome to the dark side.. we've got cookies!
securessh:/$
```

În rezultatul comenzii de mai sus se poate observa faptul ca s-au mai făcut conexiuni anterioare pe acel server de SSH, fiind oferită atât ora şi data ultimei conexiuni, cât şi adresa IP de la care s-a realizat conexiunea.

Când un client de SSH iniţiază pentru prima dată o conexiune către un server nou, acesta din urmă va trebui sa se autentifice către client. Fiecare server de SSH are un identificator unic (*host key*) cu care se autentifică clienţilor. Acest ID este implementat prin chei publice şi chei private. La primirea unei conexiuni de la un client, serverul oferă cheia sa publică, făcând astfel posibilă autentificarea. La acceptarea cheii publice de la server, clientul adaugă această cheie în fişierul `known_hosts`. Astfel, următoarea conexiune pe care acest client o va face la server nu va mai necesita transferul cheii publice, aceasta existând deja stocată pe client.

Pentru a realiza acest lucru, clientul SSH va salva cheia serverului în fişierul local `$HOME/.ssh/known_hosts`. La următoarea conectare pe acelaşi server, clientul SSH va verifica acest fişier şi, dacă va găsi cheia publică oferită de acesta se va afişa pentru autentificare doar parola.

Un exemplu de fişier `known_hosts` este următorul:

```
waters@myr:/$ ls ~/.ssh/known-hosts

acmserver,141.85.37.165                                                    ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEA7ijnAivb7dfGLkfYJlSkOwDWd2MkeP9YQctVfyb/8OfgVTLlp3eMimItJKv7rL5Angb
+A8bxd8y+tn7n0iDyoMNIAQP+rVBG2tDwlwTdl0mAhes90rOy4xOtVBOaF40dg7iy3/9zgp8HlVdiVjibuXeaIKAzew/k/I
XSB8YRd18=
atlantis.cs.pub.ro,141.85.37.4                                             ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEAqJT/wsciLHg9g1CHrFkvx9KaSl4Z7uQROWSEJd6zFWey4aMmcW//A6rrNK7DK6luv+A
eJLMHA8+1IcnzDSV+pFUH/7IeR1ryrkyGmQRjnp5crrVDPY+ixOrR3Drpn6tpEb8woW12TiOQXGNywc3g7w7VbSTP7AZGwN
lMBes26PM=
```

Material cu drept de autor

Pe prima linie sunt trecute numele serverului de la distanţă şi adresa IP a acestuia, apoi urmând cheia publică. În exemplul de mai sus există două servere la care utilizatorul s-a conectat în prealabil şi a căror cheie publică a fost salvată.

Aceste chei se pot afla în două locuri din sistemul de fişiere:

- `/etc/ssh/ssh_known_hosts`, fişier a cărui locaţie poate fi schimbată alterarea directivei `GlobalKnownHostsFile` din fişierul de configurare `/etc/ssh/ssh_config`.
- `$HOME/.ssh/known_hosts`, fişier a cărui locaţie poate fi schimbată prin alterarea directivei `UserKnownHostsFile` din acelaşi fişier de configurare.

În captura de mai jos se poate observa cum la prima conexiune realizată la un server, clientul reţine amprenta acestuia în fişierul `known_hosts`:

```
Warning: Permanently added 'localhost' (RSA) to the list of known hosts.
student@localhost's password:
Linux ubuntu 2.6.15-26-386 # 1 PREEMPT Thu Aug 3 02:52:00 UTC 2006 i686 GNU/Linux
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

6.1.2.2 Utilizarea SSH pentru rularea de comenzi la distanţă

Sintaxa completă a utilitarul `ssh` permite precizarea unei liste de parametri a utilizatorului şi a adresei destinaţie, precum şi a unei comenzi ce va fi rulată după stabilirea sesiunii SSH. Dacă nu este precizată o comandă se va rula un interpretor de comenzi (cel mai adesea `/bin/bash`).

În exemplul de mai jos, sunt rulate local două comenzi separate prin „;". Apoi se rulează aceleaşi comenzi (de data aceasta protejate între ghilimele), rezultatul afişat este cel al executării lor pe staţia destinaţie, după autentificarea cu utilizatorul `bogdand`. Se observă din exemplu că autentificarea se realizează fără a interoga utilizatorul, aceasta fiind rezultatul unei autentificări pe bază de chei.

```
waters@myr:~$ hostname; pwd
apple
/home/rrazvan
rrazvan@apple:~$ ssh bogdand@141.85.99.5 "hostname;  pwd"
kiwi
/home/users/bogdand
```

Întrebarea ce se ridică în continuare este cum se poate verifica amprenta RSA. Dacă s-a realizat o conexiune la un server SSH, dar un atacator a interceptat conexiunea, este posibil ca în urma unei neatenţii, prin simpla acceptare a respectivei chei, să se accepte de fapt cheia falsă şi nu pe cea reală. O metodă ar putea fi publicarea amprentei RSA (cheia publică) pe *site*-ul oficial al celui ce oferă un astfel de serviciu. Dacă nu există această posibilitate, cheia trebuie verificată imediat după ce v-aţi autentificat. Acest lucru se realizează folosind utilitarul `ssh-keygen` şi comparând rezultatele obţinute cu cele oferite de server. Dacă rezultatele sunt identice, atunci autentificarea s-a realizat cu succes pe staţia dorită.

```
ssh-keygen -l -f /etc/ssh/ssh_host_rsa_key.pub
```

Există două modalităţi de execuţie a comenzilor prin SSH. Prima presupune executarea comenzilor dorite după realizarea conexiunii. Prin cea de-a doua metodă comenzile dorite sunt date ca argumente clientului SSH. Pentru aceasta din urmă, de fiecare dată când se va executa o anumită comandă, va trebui introdusă parola utilizatorului ce deschide sesiunea.

```
root@myr:~#  ssh student@localhost ls
student@localhost's password:
Examples
root@myr:~#  ssh student@localhost pwd
student@localhost's password:
/home/student
root@myr:~#  ssh student@localhost uname -a
```

Material cu drept de autor

```
student@localhost's password:
Linux ubuntu 2.6.15-26-386 # 1 PREEMPT Thu Aug 3 02:52:00 UTC 2006 i686 GNU/Linux
```

6.1.2.3 Fişierul de configurare pentru server

Fişierul principal pentru configurarea serverului SSH este `/etc/ssh/sshd_config`. Câteva directive importante pentru configurarea serviciului sunt:

- `[Port 22]` – specifică portul pe care ascultă serverul; portul implicit este 22.
- `[ListenAddress 192.168.1.2]` – specifică interfaţa pe care ascultă *daemon*-ul `sshd`. Adresa implicită este 0.0.0.0, reprezentând faptul că se ascultă pe toate interfeţele.
- `[HostKey /etc/ssh/ssh_host_key]` – specifică fişierul în care este ţinută cheia privată a utilizatorului.
- `[ServerKeyBits 1024]` – defineşte numărul de biţi din *server key* (implicit 768).
- `[KeyRegenerationInterval 3600)` – specifică după cât timp va fi regenerată cheia serverului. Aceasta poate fi o metodă de a preveni decriptarea în cazul interceptării unei sesiuni în urma unui atac *man-in-the-middle*.
- `[PermitRootLogin]` – poate fi „yes", „nopwd" sau „no" şi se referă la posibilitatea autentificării prin SSH folosind contul de `root`; „nopwd" semnifică faptul că nu este permisă autentificarea cu parolă. Această opţiune trebuie întotdeauna setată pe „no" din motive de securitate.
- `[X11Forwarding yes]` – specifică dacă este permisă redirectarea conexiunilor X11 peste o conexiune de SSH.
- `[RSAAuthentication yes]` – specifică folosirea autentificării folosind protocolul RSA.
- `[AllowUsers admin]` – specifică utilizatorii care se pot conecta prin acest serviciu.
- `[PrintMotd]` – specifică dacă `sshd`-ul va afişa conţinutul fişierului `/etc/motd` după autentificarea unui utilizator. MOTD (*Message of the Day*) reprezintă un text afişat utilizatorului după autentificare, dar înainte de apariţia promptului, care conţine mesaje de la administrator.

6.1.2.4 Copierea fişierelor la distanţă

Unul dintre utilitarele importante ce aparţin pachetului OpenSSH este `scp` (*Secure Copy*). SCP este un protocol folosit în transferul sigur de fişiere între două staţii, pe durata transferului datele fiind criptate de o sesiune SSH. Protocolul în sine nu asigură autentificarea şi securitatea comunicaţiei, acestea fiind asigurate de către protocolul SSH.

Astfel, dacă se doreşte copierea unui fişier de pe staţia locală pe o staţie la distanţă se poate folosi următoarea sintaxă:

```
scp cale_fişier_local utilizator@staţie_distanţă:/cale_fişier
```

În schimb, dacă se doreşte copierea unui fişier de pe o staţie la distanţă pe staţia locală se poate folosi următoarea sintaxă:

```
scp utilizator@staţie_distanţă:/cale_ffşier_remore /cale_fişier
```

Notă: *daca se foloseşte o adresare relativă, nu se mai foloseşte „/" la începutul căii fişierului*

6.1.2.4.1 Copierea unui singur fişier pe o staţie la distanţă

În exemplul de mai jos, se va copia fişierul `test.txt` de pe staţia locală (directorul local) pe staţia `anaconda.cs.pub.ro`. Fişierul va fi copiat în directorul `home` al utilizatorului `user`, cu numele `NewTest.txt`.

```
root@myr:~# scp rest.txt user@anaconda.cs.pub.ro:NewTest.txt
```

Material cu drept de autor

Pentru a păstra numele original al fişierului ce trebuie copiat se omite specificarea noului nume.

```
root@myr:~# scp test.txt user@anaconda.cs.pub.ro:
```

Într-un mod asemănător se poate realiza copierea unui director şi a conţinutului acestuia:

```
root@myr:~# scp -r test user@anaconda.cs.pub.ro:Data/NewTest/
```

Folosind opţiunea `-r`, directorul local `test` va fi copiat recursiv în directorul `Data/NewTest`, aflat în directorul `home` al utilizatorului `user` de pe staţia `anaconda.cs.pub.ro`.

6.1.2.4.2 Copierea unui singur fişier de pe o staţie la distanţă

Pentru copierea fişierului `data.txt` de pe staţia `anaconda.cs.pub.ro`, din directorul `home` al utilizatorului *user*, în directorul curent putem folosi următoarea sintaxă:

```
root@myr:~# scp user@anaconda.cs.pub.ro:data.txt NewData.txt
```

În comanda de mai sus, fişierul se salvează în directorul curent sub numele de `NewData.txt`.

Cum s-a menţionat şi mai sus, se poate specifica şi calea pentru fişierul ce se doreşte a fi copiat. În exemplul de mai jos, fişierul `data.txt`, aflat în subdirectorul `scoala/teste` (relativ la directorul `home` al utilizatorului *user*), va fi copiat în directorul curent, fişierul păstrând acelaşi nume.

```
root@myr:~# scp user@anaconda.cs.pub.ro:scoala/teste/data.txt .
```

6.1.3 Configuraţii avansate SSH

6.1.3.1 Redirectarea X şi TCP/IP peste SSH

Dacă utilizatorul foloseşte X11 (variabila de mediu `DISPLAY` fiind setată), conexiunea cu *display*-ul X11 este transferată automat la distanţă în aşa fel încât orice program X11 pornit din shell (sau printr-o comandă) este trecut prin canalul criptat şi conexiunea cu adevăratul server X va fi făcută de pe maşina locală. Utilizatorul nu trebuie să seteze manual variabila `DISPLAY`. Transferarea conexiunilor X11 poate fi configurată din linia de comandă sau din fişierele de configurare.

Redirectarea protocolului X11 trebuie activată atât în fişerul `sshd_config` (prin variabila `X11Forwarding`) cât şi în fişierul `ssh_config` (prin variabila `ForwardX11`). Ambele variabile trebuiesc setate la valoarea „`yes`".

În linia de comandă putem folosi opţiunea `-X` pentru stabilirea conexiunii.

```
root@myr:~# ssh -X student@anaconda.cs.pub.ro
```

Transferarea unei conexiuni TCP/IP prin canalul sigur oferit de SSH poate fi specificată fie din linia de comandă, fie din fişierele de configurare. O aplicaţie posibilă a redirectării TCP/IP este trecerea de un *firewall* în vederea citirii poştei electronice.

Utilitarul `ssh` menţine şi verifică automat o bază de date cu identificările bazate pe RSA ale tuturor maşinilor pe care utilizatorul s-a conectat. Baza de date este ţinută în `.ssh/known_hosts`. Dacă informaţia de identificare a unui server se schimbă, `ssh` afişează un avertisment şi nu permite autentificarea utilizatorului pentru a preveni un atentat la parola lui. Opţiunea `StrictHostKeyChecking` poate fi folosită pentru a preveni autentificările pe maşini ale căror chei nu sunt cunoscute sau care au fost schimbate.

Material cu drept de autor

6.1.3.2 Generarea cheilor SSH

Aşa cum se observă şi în etapa de instalare a serverului, este necesară generarea unei perechi de chei (publică/privată). În cazul în care se doreşte să se creeze o nouă pereche de chei, se foloseşte utilitarul `ssh-keygen`. Cu ajutorul acestui utilitar se creează două fişiere, unul pentru fiecare cheie. Implicit, în fişierul `~/.ssh/id_rsa` este păstrată cheia privată, iar în fişierul `~/.ssh/id_rsa_pub` cheia publică. Deoarece cheia privată nu trebuie ţinută la vedere, `ssh-keygen` oferă posibilitatea introducerii unei parole („*passphrase*"), ce va proteja fişierul în care se păstrează aceasta.

```
root@myr:~#  ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
bc:14:cc:41:ca:bc:b6:a2:10:63:19:cd:fc:68:da:32 root@ubuntu
root@ubuntu:~#
```

În cazul în care se doreşte schimbarea *passphrase*-ului se poate folosi opţiunea `-p`:

```
root@myr:~#  ssh-keygen -p
Enter file in which the key is (/root/.ssh/id_rsa):
Enter old passphrase:
Key has comment '/root/.ssh/id_rsa'
Enter new passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved with the new passphrase.
```

6.1.3.3 Conectare fără parolă

Dacă se deschid sesiuni SSH frecvente cu diferite staţii ce oferă autentificare prin chei, se poate folosi utilitarul `ssh-agent`. Acesta se ocupă de gestionarea şi stocarea cheilor private. Pentru conectarea fără parolă pe serverul la distanţă, utilizatorul trebuie să îşi copieze cheia publică în fişierul `$HOME/.ssh/authorized_keys`, aflat pe server.

```
root@myr:~#  ssh-copy-id -i /root/.ssh/id_rsa.pub student@localhost
21
student@localhost's password:
Now try logging into the machine, with "ssh 'student@localhost'", and check in:
  .ssh/authorized_keys
to make sure we haven't added extra keys that you weren't expecting.
```

După ce se realizează acest lucru, trebuie activat serviciul de stocare a cheilor private (`ssh-agent`) şi apoi adăugate cheile în acesta (`ssh-add`). Pentru adăugarea cheilor private va trebui introdus *passphrase*-ul fiecăreia, astfel încât accesul să fie permis pentru adăugarea lor în fişierul `~/.ssh/id_rsa`. În urma acestor paşi, conectarea se va face automat, fără nevoia introducerii unei parole.

```
root@myr:~#  ssh-agent
SSH_AUTH_SOCK=/tmp/ssh-nWcDud7170/agent.7170; export SSH_AUTH_SOCK;
SSH_AGENT_PID=7171; export SSH_AGENT_PID;
echo Agent pid 7171;
root@ubuntu:~#  ssh-add
Enter passphrase for /root/.ssh/id_rsa:
Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
```

```
root@myr:~#  ssh student@localhost
Linux ubuntu 2.6.15-26-386 # 1 PREEMPT Thu Aug 3 02:52:00 UTC 2006 i686 GNU/Linux
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
Last login: Tue Sep 18 14:55:00 2007 from localhost
```

Material cu drept de autor

6.1.3.4 Tunelare trafic printr-o conexiune SSH

Tunelarea SSH reprezintă folosirea protocolului SSH şi a capabilităţilor acestuia, pentru a securiza un transfer de date. Pentru crearea unui tunel SSH, clientul SSH este configurat să redirecteze un port specificat şi o adresă IP (existente pe serverul SSH la distanţă) pe un port al staţiei locale. Odată ce conexiunea SSH a fost stabilită, utilizatorul se poate conecta la portul local pentru accesarea serviciilor aflate pe staţia la distanţă (pe serverul SSH), servicii ce altfel ar fi putut fi accesibile prin conectarea direct pe portul şi adresa IP a staţiei la distanţă. Cu alte cuvinte, această metodă poate fi văzută ca o încercare de ocolire a *firewall*-urilor, ce filtrează anumite servicii Internet.

Pentru a ilustra cât mai bine cele menţionate mai sus se consideră următorul scenariu. Un utilizator obişnuit doreşte să îşi citească email-ul, folosind un client de email (*Thunderbird*, *fetchmail*, *mutt*, *Outlook* etc.). Dacă utilizatorul se conectează direct la serverul de e-mail, clientul de e-mail va trimite numele de utilizator şi parola în text clar. Acest lucru reprezintă o mare vulnerabilitate în cazul în care un intrus ascultă cu un *sniffer* mediul.

Pentru prevenirea acestui atac se pot folosi capabilităţile de tunelare oferite de SSH. Un tunel SSH va fi folosit în modul următor: în loc să se realizeze conectarea direct la serverul de e-mail, se va stabili o conexiune SSH către un server din interiorul reţelei în care respectivul server de e-mail se află, sau direct către serverul însuşi (aşa cum se întâmplă cel mai adesea, când serverul de e-mail oferă şi serviciu SSH). Clientul SSH va realiza redirectarea portului, astfel încât traficul ce ajunge la staţia locală pe portul 110 (portul implicit folosit de clientul de email), ajunge să fie redirectat prin tunelul criptat către serverul de e-mail. Apoi clientul de email poate fi configurat să utilizeze portul local, fiind pus astfe în legătură cu serverul de la distanţă.

Exemplul următor ilustrează snecariul discutat:

```
root@myr:~# ssh -L 110:mailhost:110 -l user -N mailhost
```

mailhost reprezintă numele serverului de email, 110 este portul implicit pe care ascultă clientul de email, *user* este numele de utilizator şi opţiunea `-N` specifică doar redirectarea portului, fără executarea de comenzi.

Un alt exemplu poate fi următorul:

```
root@myr:~# ssh -L 7777:work:22 -l user gate
```

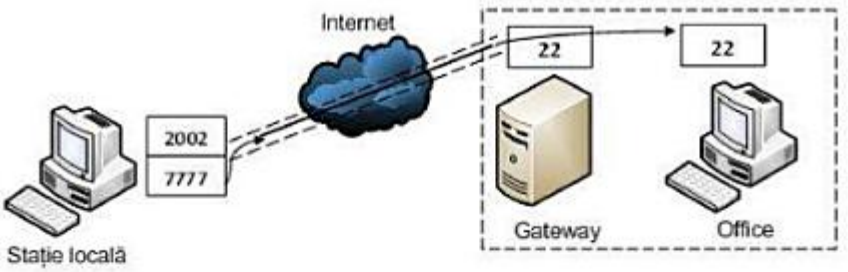
Regula de mai sus poate fi interpretată astfel: se deschide o sesiune SSH a utilizatorului *user*, pe staţia *gate*. Cât timp sesiunea rămâne deschisă, toate conexiunile către portul 7777 de pe staţia locală sunt redirectate către portul 22 de pe staţia `work`.

Pentru conectarea prin tunel se specifică portul local 7777.

```
root@myr:~# ssh -p 7777 localhost
```

Opţiunea `-L` va spune staţiei *gate* că de fiecare dată când va primi date venind cu un marcaj de tunel, să deschidă o sesiune pe portul 22 pe staţia *work* şi să trimită respectivele date. Toate datele sunt transmise staţiei *gate* cu un port sursă aleator (ex: 2002) şi cu un marcaj de tunel. Cu alte cuvinte, staţia locală acceptă conexiuni pe portul 7777 şi trimite toate datele pe portul 2002 staţiei *gate*, spunându-i că acestea provin de la „tunelul 7777".

Material cu drept de autor



6-2: Tunelare SSH

Pentru a exemplifica lucrurile menţionate până acum, în secţiunea de mai jos este prezentat modul în care se poate realiza un tunel SSH.

```
root@myr:~#  ssh -L 7777:anaconda.cs.pub.ro:22 -l cico anaconda.cs.pub.ro
Password:
Last login: Sat Sep 22 13:30:01 2007 from 86.120.171.63
Linux anaconda 2.6.18-4-686 # 1 SMP Wed May 9 23:03:12 UTC 2007 i686
[....]
anaconda:~#
```

Astfel se realizează deschiderea tunelului pe staţia `anaconda.cs.pub.ro`. Toate conexiunile acum vor fi redirectate prin portul 7777. Se verifică şi deshiderea acestuia pe staţia locală.

```
root@myr:/etc/ssh#  nmap -p 7775-7778 localhost

Starting Nmap 4.03 (http://www.insecure.org/nmap/) at 2007-09-22 14:03EEST
Interesting ports on localhost (127.0.0.1):
PORT     STATE  SERVICE
7775/tcp closed unknown
7776/tcp closed unknown
7777/tcp open   unknown
7778/tcp closed unknown

Nmap finished: 1 IP address (1 host up) scanned in 0.057 seconds
```

După ce sa confirmat că portul 7777 a fost deschis, se poate realiza o conexiune prin tunelul SSH deschis.

```
root@myr:/root/.ssh#  ssh -p 7777 -l cico localhost
Password:
Last login: Sat Sep 22 13:58:01 2007 from 86.121.174.62
Linux anaconda 2.6.18-4-686 # 1 SMP Wed May 9 23:03:12 UTC 2007 i686
[....]
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
```

6.2 *Firewall*

Probabil că cel mai cunoscut dispozitiv de securitate este *firewall*-ul. Prin definiţie, *firewall*-ul este un sistem sau un grup de sisteme care implementează politica de acces între două sau mai multe reţele. *Firewall*-urile pot fi clasificate în patru mari clase: *firewall*-uri dedicate, *firewall*-uri integrate în routere, *firewall*-uri integrate în servere şi *firewall*-uri personale.

Firewall-urile dedicate sunt maşini ce rulează un sistem de operare special conceput pentru filtrarea de pachete şi translatarea de adrese. Exemple de astfel de *firewall*-uri sunt sistemele PIX sau *CheckPoint*. Aceste sisteme sunt capabile să susţină un număr mare de conexiuni, dar facilităţile de rutare sunt extrem de limitate. Pentru o reţea simplă se poate folosi *firewall*-ul ca router, însă pentru reţele mai complexe este necesar un router.

Firewall-urile integrate în routere sunt folosite pentru a înlătura neajunsul anterior. Ele nu pot susţine acelaşi număr de conexiuni, dar se descurcă mai bine în topologii mai complexe,

Material cu drept de autor

unde este nevoie de facilitățile unui router. Multe produse oferă facilități de *firewall* integrate în routere, de la module de *firewall* pentru routere *high-end*, până la routerele extrem de compacte, dedicate utilizării în rețele SOHO[1].

Firewall-urile de server sunt implementate ca un software adițional peste un sistem de operare de rețea (Linux, NT, Win2K, Novell, UNIX). Exemple de astfel de pachete software sunt: *Netfilter*, *Microsoft ISA Server*, *Novell Border Manager*. Ele sunt comparabile ca facilități și performanțe cu *firewall*-urile integrate în routerele de nivel mediu.

Firewall-urile personale sunt instalate pe calculatoarele personale. Ele sunt concepute pentru a preveni atacuri doar asupra calculatorului pe care rulează. Este important de reținut că aceste tipuri de *firewall*-uri nu sunt optimizate pentru rețele întregi de calculatoare. Exemple de firme ce produc *firewall*-uri personale sunt *McAfee* și *Symantec*.

Principalele mecanisme prin care un *firewall* asigură protecția rețelei sunt filtrarea de pachete și translatarea de adrese, care vor fi analizate mai pe larg în continuare.

6.2.1 Filtrarea de pachete

Filtrarea de pachete este procesul prin care doar anumite pachete sunt rutate dintr-o rețea în alta, pe baza unor reguli. Filtrarea de pachete operează tradițional cu informații de la nivelurile OSI 3 și 4.

Regulile de filtrare sunt formate dintr-o parte care identifică pachetul și o parte care specifică cum să se trateze pachetul. În partea de identificare se poate specifica adresa sursă, adresa destinație, adresa de rețea sursă, adresa de rețea destinație, protocolul (TCP, UDP, ICMP), portul sursă sau destinație (doar pentru TCP sau UDP), tipul mesajului (pentru ICMP), interfața de intrare sau ieșire, și chiar și adresele de nivel doi. În principiu se poate face identificarea pachetului cu orice informație scrisă în antetul pachetului, la nivelul OSI 3, 4 sau chiar și 2, în funcție de implementare.

Partea de tratare a pachetului specifică ce anume trebuie făcut cu pachetele selectate de o regulă. Pentru filtrare există în general trei posibilități de tratare: acceptare, ignorare sau respingere. În cazul acceptării, pachetul este lăsat să treacă. În cazul ignorării pachetului nu este lăsat să treacă și nu se trimite notificare către sursă. În cazul respingerii, pachetul nu este lăsat să treacă, dar se trimite notificare către sursă (un mesaj ICMP al cărui tip poate fi, în unele implementări, ales de cel care construiește regula; de cele mai multe ori se folosește un mesaj ICMP de tip `port-unreachable`).

6.2.1.1 Utilitarul iptables

Iptables este utilitarul cu ajutorul căruia se pot configura politica și regulile de filtrare de pachete sau translatare de adrese pentru Linux 2.4. Acesta face parte din proiectul *Netfilter*, care implementează în Linux filtrarea de pachete și translatarea de adrese.

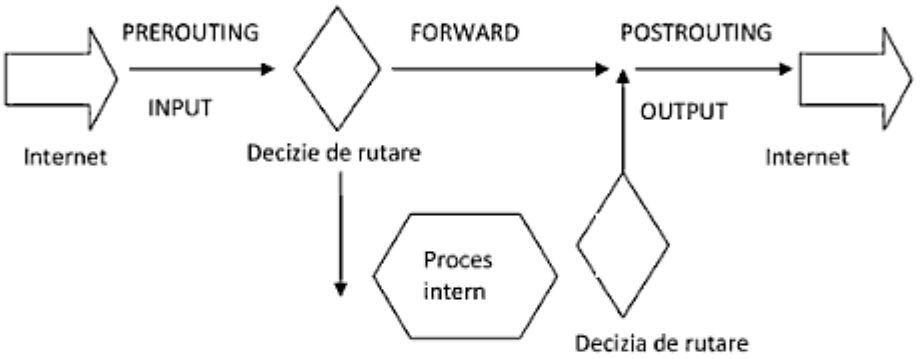
În `iptables` o regulă are două părți: o parte care identifică pachetele și una care specifică cum trebuie tratate pachetele respective (partea țintă). Procesarea regulilor se face secvențial începând cu prima regulă. Dacă pentru un pachet ce traversează sistemul regula curentă este validă se va execută acțiunea asociată țintei. Dacă nu, se trece la următoarea regulă. Dacă s-au epuizat toate regulile dintr-un lanț definit de utilizator sau dacă ținta este `RETURN`, se continuă analizarea regulilor din lanțul precedent. Dacă s-au epuizat toate regulile dintr-un lanț predefinit, se execută acțiunea asociată politicii implicite a lanțului.

[1] Small Office Home Office

Material cu drept de autor

Pachetul poate fi identificat după adresa sursă, adresa destinație, tipul pachetului, portul (TCP, UDP) sau tipul mesajului (ICMP), interfața pe care intră/iese pachetul, dacă este fragment dintr-un pachet, dacă este pachet care inițiază o conexiune (TCP).

Lanțurile sunt seturi de reguli prin intermediul cărora se determină ce acțiune trebuie luată asupra unui pachet. Pentru fiecare dintre tabelele definite (`filter`, `nat`, `mangle`) există lanțuri implicite (`INPUT`, `OUTPUT`, `FORWARD`, `PREROUTING`, `POSTROUTING`) ce asigură o structură distribuită a regulilor. Cu alte cuvinte, lanțurile predefinite nu caracterizează o singură tabelă, tabelele împărțind unul sau mai multe lanțuri. De exemplu, lanțul de `OUTPUT` aparține atât tabelei `filter` cât şi tabelei `nat`. La fel, lanțul de `INPUT` aparține atât tabelei `filter` cât şi tabelei `mangle`. Când un pachet ajunge la o stație ce implementează o astfel de politică (bazată pe iptables), vor trebui luate anumite decizii asupra acestuia, realizându-se analiza fiecărui lanț menționat mai sus. În figura de mai jos se ilustrează modul în care se va face verificarea fiecărui lanț.



6-3: Fluxul pachetelor prin lanțurile predefinite

	PREROUTING	INPUT	FORWARD	OUTPUT	POSTROUTING
nat	X			X	X
mangle	X	X	X	X	X
filter		X	X	X	

6-4: Asocierea dintre tabele şi lanțuri la *iptables*

6.2.1.2 Filtrarea de pachete folosind iptables

Pentru filtrarea de pachete se foloseşte **tabela filter**. Pentru această tabelă există trei lanțuri predefinite: `INPUT` - pachete ce sunt destinate routerului, `OUTPUT` - pachete generate de router, şi `FORWARD` - pachete care sunt rutate (pachete care nici nu sunt generate de router, nici nu sunt destinate routerului).

Țintele ce pot fi folosite sunt `ACCEPT` - pachetele sunt lăsate sa treacă, `DROP` - pachetele sunt ignorate, `QUEUE` - pachetele sunt copiate în *user-space* pentru analize, şi `LOG` - pachetele sunt scrise în log.

Se poate folosi de asemenea şi `REJECT` pentru a respinge pachetele. La această țintă se poate specifica tipul mesajului `icmp` folosit pentru notificare cu opțiunea `--reject-with`.

Pentru a ilustra mai bine lucrurile menționate mai sus fie următoarea regulă:

```
iptables -t filter -A INPUT -s 10.0.0.0/8 -p icmp -j DROP
```

Material cu drept de autor

Regula de mai sus poate fi interpretată în modul următor: toate pachetele ce sunt destinate stației locale, ce au adresa IP sursă în rețeaua 10.0.0.0/8 și sunt de tip ICMP, nu vor fi acceptate. Opțiunea `-t` precizează tabela, în cazul de față `filter`, -A (*append*) specifică adăugarea regulii lanțului de `INPUT`, `-s` specifică adresa IP sursă a pachetelor, `-p` protocolul folosit, iar `-j` (*jump*) precizează acțiunea ce va trebui îndeplinită în cazul în care pachetele se încadrează în regula respectivă.

În exemplul de mai jos este prezentată configurarea `iptables` pentru filtrare de pachete pe mașina `141.85.37.1` din rețeaua `141.85.37.0/24`. Această mașină este router, *firewall*, server de web, server de e-mail și server de DNS și are conectată rețeaua internă pe interfața `eth0` și legătura la Internet pe interfața `eth1`.

```
iptables -t filter -A INPUT -s 192.168.0.0/16 -j DROP
iptables -t filter -A INPUT -s 10.0.0.0/8 -j DROP
iptables -t filter -A INPUT -p tcp --destination-port 22 -j ACCEPT
iptables -t filter -A INPUT -p tcp --destination-port 80 -j ACCEPT
iptables -t filter -A INPUT -p tcp --destination-port 25 -j ACCEPT
iptables -t filter -A INPUT -p tcp --destination-port 53 -j ACCEPT
iptables -t filter -A INPUT -p udp --destination-port 53 -j ACCEPT
iptables -t filter -A INPUT -p tcp ! --syn -j ACCEPT
iptables -t filter -A INPUT -j REJECT

iptables -t filter -A FORWARD -s 192.168.0.0/16 -j DROP
iptables -t filter -A FORWARD -s 10.0.0.0/8 -j DROP
iptables -t filter -A FORWARD -i eth1 -s 141.85.37.0/24 -j ACCEPT
iptables -t filter -A FORWARD -i eth0 -p tcp ! --syn -j ACCEPT
iptables -t filter -A FORWARD - j REJECT

iptables -t filter -A OUTPUT -s 192.168.0.0/16 -j DROP
iptables -t filter -A OUTPUT -s 10.0.0.0/8 -j DROP
```

Configurarea pentru pachetele ce sunt destinate mașinii (lanțul `INPUT`) conține reguli pentru filtrarea pachetelor cu adrese private (pentru a evita atacurile ce folosesc falsificare adreselor IP), iar apoi reguli pentru accesarea serviciilor ce rulează pe mașină: serverul de ssh, serverul de web, serverul de e-mail, serverul de DNS. Penultima regulă este necesară dacă dorim să putem accesa de pe server Internetul sau rețeaua locală. Regula va opri încercările de conectare la server, dar va lăsa pachetele de răspuns să se întoarcă la server.

Configurarea pentru pachetele ce trebuie rutate (lanțul `FORWARD`) începe cu două reguli pentru filtrarea pachetelor cu adrese private. Următoarea regulă acceptă toate pachetele ce vin din rețeaua internă și au adrese corecte (pentru a preveni atacuri ce folosesc falsificarea adresei IP inițiate din interiorul rețelei). Următoarea regulă acceptă pachetele din Internet ce sunt replici la traficul inițiat din interior, dar nu și pachetele de inițiere a unor conexiuni către rețeaua internă.

În fine, configurarea pentru pachetele generate de server conține reguli pentru filtrarea adreselor private, pentru a evita folosirea serverului în atacuri ce falsifică adresa IP.

6.2.2 Translatarea de adrese

Translatarea de adrese sau NAT (*Network Address Translation*) este procesul prin care se modifică adresele sursă (SNAT) sau destinație (DNAT) din anumite pachete care trec prin *firewall*. Din punctul de vedere al securității, translatarea de adrese se folosește pentru a ascunde modul de adresare intern și pentru a evita accesarea stațiilor interne din exterior, prin folosirea unor adrese private în rețeaua internă, adrese ce vor fi translatate la adrese publice pentru ca stațiile interne să aibă acces la Internet.

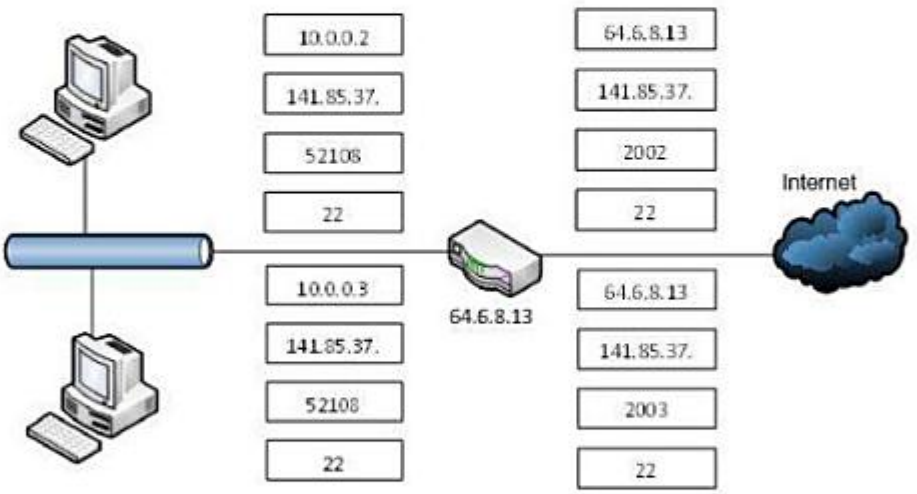
Putem considera că translatarea adreselor este o funcție definită pe o mulțime de adrese (A) cu rezultate într-o altă mulțime de adrese (B). Astfel, fiecare pachet cu o adresă sursă sau destinație din mulțimea A va fi înlocuită cu o adresă din mulțimea B. Se spune că se realizează o **translatare de adrese statică** dacă funcția de translatare este injectivă, adică fiecărei adrese din mulțimea A îi corespunde o singură adresă corespunzătoare în mulțimea B. Dacă funcția de

Material cu drept de autor

translatare nu este injectivă, adică dacă pentru mai multe adrese din mulţimea A corespunde o singură adresă din mulţimea B, aceasta poartă denumirea de **translatare de adrese dinamică**.

Avantajul folosirii translatării de adrese dinamice constă în faptul că se pot partaja adrese rutabile disponibile. Astfel, calculatoarelor din reţeaua locală li se alocă adrese private, iar routerul (*firewall*-ul) va face o translatare de adrese dinamică din mulţimea de adrese private în mulţimea de adrese publice. Se observă însă că această abordare permite ca doar Card(B) calculatoare din reţeaua locală să aibă conversaţii simultane în Internet.

O metodă mai avansată de translatare de adrese o reprezintă translatarea de adrese supraîncărcată **NAT overloaded,** denumită uneori şi **PAT** (*Port Address Translation*) sau *masquerading*. Această metodă permite un număr de aproximativ 64000 de conversaţii simultane de la orice *host* intern către exterior cu o singură adresă externă. Implementarea înlocuieşte pachetul din reţeaua locală cu adresa sursă S, adresa destinaţie D, portul sursă P, portul destinaţie Q, cu altul nou ce va avea adresa sursă M (adresa routerului), adresa destinaţie D, portul sursă K. Portul destinaţie nu se schimbă. De asemenea se memorează asocierea (S,P) - K. Dacă un pachet ajunge pe router din exterior, având adresa destinaţie M, adresa sursă Q şi portul destinaţie K, atunci acest pachet va fi înlocuit cu un altul cu adresa destinaţie S, adresa sursă Q, portul destinaţie P şi va fi trimis în reţeaua locală. Portul sursă nu se schimbă.



6-5: NAT overloaded (PAT)

Un caz special al PAT îl reprezintă **redirectarea**. În acest caz se va înlocui pachetul primit din reţeaua locală având adresa sursă S, adresa destinaţie D, portul P cu un altul având adresa sursă S, adresa destinaţie M (adresa routerului), portul R (portul în care se face redirectarea, specificat de utilizator). Redirectarea este în general folosită pentru a implementa un *proxy* transparent, caz în care pe routerul M portul R ascultă un *proxy* configurat pentru *proxy* transparent.

6.2.2.1 Translatarea de adrese folosind iptables

Pentru translatarea de adrese se foloseşte **tabela nat**. În această tabelă există trei lanţuri predefinite: `PREROUTING` - modifică pachetul imediat ce acesta intră în router, înainte de a fi rutat, `OUTPUT` - modifică pachetele generate local înainte ca acestea să intre în procesul de

Material cu drept de autor

rutare, şi POSTROUTING - modifică pachetele ce urmează să plece din router, după ce acestea au fost rutate. Ţintele valide sunt ACCEPT, DROP, QUEUE, REJECT, LOG, SNAT, DNAT, MASQUARADE, REDIRECT.

SNAT se foloseşte pentru a indica o translatare de adrese de tip PAT pe adresa sursă. Adresa sursă a pachetului va fi modificată la una din intervalul specificat prin opţiunea --to-source. Cu aceeaşi opţiune se poate specifica şi intervalul în care se va alege portul sursă când se face translatarea de adrese. Această ţintă este validă numai în lanţul POSTROUTING (şi lanţurile chemate din acest lanţ).

DNAT se foloseşte pentru o translatare de adrese de tip PAT pe adresa destinaţie. Adresa destinaţie a pachetului va fi modificată la una din intervalul specificat prin opţiunea --to-destination. Această ţintă este validă numai în lanţurile PREROUTING şi OUTPUT (şi lanţurile chemate din acest lanţ).

MASQUERADE este echivalent cu SNAT. Adresa sursă va fi înlocuită cu adresa setată a interfeţei pe care va fi trimis pachetul. Trebuie folosită în loc de SNAT dacă adresa la care se face translatarea este setată dinamic (prin DHCP de exemplu).

REDIRECT se foloseşte pentru a redirecta pachetul, local, pe portul specificat de opţiunea --to-port. Această ţintă este validă numai în lanţurile PREROUTING şi OUTPUT.

Pentru a evidenţia mai bine lucrurile menţionate mai sus pot fi analizate următoarele două reguli:

```
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.0.0/24 -j SNAT   --to-source 1.2.3.4
iptables -t nat -A PREROUTING -i eth0 -d 14.15.16.17 -j DNAT  --to-destination 192.168.100.1
```

Prima regulă poate fi interpretată în felul următor: toate pachetele ce vin cu adresa IP sursă din reţeaua 192.168.0.0/24 vor fi trimise pe interfaţa eth1 cu adresa IP sursă 1.2.3.4, după ce acestea vor fi rutate. Cea de-a doua regulă va schimba adresa IP destinaţie (14.15.16.17) a pachetelor ce intră pe interfaţa eth0 cu adresa IP 192.168.100.1, înainte ca acestea să fie rutate.

În exemplul de mai jos s-a prezentat configurarea iptables pentru translatarea de adrese pe maşina 141.85.37.1 din reţeaua 192.168.1.0/24. Această maşină este doar router, iar serverul de web, serverul de mail şi serverul de DNS rulează pe servere diferite, în reţeaua internă. Routerul are conectată interfaţa eth0 la reţeaua internă şi interfaţa eth1 la Internet.

```
        iptables -t nat -A POSTROUTING -o eth1 -s 192.168.1.0/24 -j SNAT --to-source 141.85.37.1

        iptables -t nat -A PREROUTING -i eth1 -p tcp --destination-port 80 -j DNAT --to-
destination 192.168.1.2
        iptables -t nat -A PREROUTING -i eth1 -p tcp --destination-port 25 -j DNAT --to-
destination 192.168.1.3
        iptables -t nat -A PREROUTING -i eth1 -p tcp --destination-port 53 -j DNAT --to-
destination 192.168.1.4
        iptables -t nat -A PREROUTING -i eth1 -p udp --destination-port 53 -j DNAT --to-
destination 192.168.1.4
```

Prima linie din exemplu este folosită pentru ca staţiile din reţea să poată accesa Internetul. La trecerea pachetelor prin router, adresa sursă a staţiilor (adresă privată) va fi înlocuită cu adresa routerului (adresă publică).

Următoarele linii vor trimite traficul de web, e-mail şi DNS către serverele din interior. În acest context, din exterior, aparent routerul este şi server de web, e-mail şi DNS.

6.2.3 Configurări avansate iptables

6.2.3.1 Port forwarding

Ce se întâmplă dacă dorim ca serviciile din reţeaua internă, ce sunt făcute publice, să fie văzute din exterior ca rulând pe porturi diferite faţă de cele standard? Metoda ce poate fi aplicată aici se numeşte **port forwarding**. Această metodă permite unei staţii (*firewall*) să

Material cu drept de autor

trimită cererile ce îi sunt adresate, altei staţii ce va procesa aceste cereri. Cel mai folosit mod de întrebuinţare al acestei metode este acela când serverele rulează pe staţii aflate în reţeaua internă (după *firewall*). Cu alte cuvinte, se presupune existenţa unui gateway cu două interfeţe, `eth0` fiind conectată la reţeaua internă şi eth1 la Internet. Fie 141.85.37.1 adresa IP publică a interfeţei `eth1` şi 192.168.0.2 adresa IP a staţiei pe care rulează un serviciu web, pe portul implicit 80, staţie ce va putea fi accesată din exterior. În exemplul de mai jos se va face redirectarea conexiunilor ce vin pe 141.85.37.1:8888 (<IP_extern:port>) către 192.168.0.2:80 (IP_intern:port).

```
iptables -t nat -A PREROUTING -i eth1 -p tcp -d 141.85.37.1 -dport 8888 -j DNAT --to-
destination 192.168.0.2:80
```

6.2.3.2 Alterarea avansată a pachetelor

Tabela mangle este folosită pentru a modifica pachetele într-un mod mai special. NAT modifică doar adresele dintr-un pachet. Tabela *mangle* poate fi folosită pentru a schimba informaţii precum TTL, TOS, sau pentru a marca un pachet. Marcarea pachetelor este folosită doar în interiorul routerului. Odată ce un pachet părăseşte routerul, informaţiile adăugate la marcare vor fi îndepărtate. În prezent marcarea pachetelor este folosită de către sistemul de *Quality of Service* (QoS).

Există trei lanţuri predefinite: `INPUT` - modifică pachetele destinate routerului, `FORWARD` - modifică pachetele în curs de rutare, şi `POSTROUTING` - modifică pachetele după rutare. Ţintele valide includ `ACCEPT`, `DROP`, `QUEUE`, `REJECT`, `LOG` precum şi cele specifice tabelei, prezentate mai jos.

`MARK` marchează pachetul cu valoarea specificată prin opţiunea `--set-mark`. Pachetele marcate se pot folosi ulterior în procesul de rutare sau QoS.

`TOS` setează câmpul de *type of service* la valoarea specificată prin opţiunea `--set-tos`.

`TTL` setează câmpul TTL la valoarea specificată prin opţiunea `--ttl-set`, decrementează valoarea acestuia (dacă se foloseşte opţiunea `--ttl-dec`) sau incrementează valoarea acestuia (dacă se foloseşte opţiunea `--ttl-inc`).

În exemplul de mai jos este prezentat un exemplu de alterare avansată a pachetelor, în care două staţii din reţeaua internă, `141.85.37.13` şi `141.85.37.169` pot accesa doar reţeaua internă (reţeaua departamentului) şi reţeaua imediat următoare (reţeaua organizaţiei).

```
iptables -t mangle -A FORWARD -s 141.85.37.13 -j TTL --ttl-set 2
iptables -t mangle -A FORWARD -s 141.85.37.169 -j TTL --ttl-set 2
```

6.3 Capturare pachetelor şi analiza pachetelor. IDS/IPS.

Termenul de *sniffing* este unul din cele mai populare noţiuni folosite în lumea administrării reţelelor.

> ***Sniffing** este procesul prin care se capturează şi se analizează traficul ce traversează segmentul local*

Utilitarele folosite pentru captura de pachete au evoluat foarte mult de la începuturile Internetului până astăzi. Plecând de la capturarea în linie de comandă, s-au dezolvoltat aplicaţii complexe în GUI, care prezintă conţinutul pachetelor capturate într-o structura bine gândită si uşor de interpretat. Una din aceste aplicaţii se numeşte *Wireshark*.

6.3.1 Wireshark – configurări de bază

Wireshark este o aplicaţie gratuită realizată în cadrului unui proiect *open-source*, a cărui rol principal este captura şi analiza pachetelor din reţea. Majoritatea administratorilor de reţea cunosc *Wireshark* sub numele de *Ethereal*, pentru că până în anul 2006 acesta era de

Material cu drept de autor

fapt numele atribuit utilitarului. Trebuie specificat că nu se urmăreşte niciun fel de exagerare în afirmaţia faptului că *Wireshark* este considerat mâna dreapta a oricărui administrator. Cu ajutorul său se poate face inspecţia la nivelul antentelor unui pachet şi mai mult la nivel de *stream* TCP. Aplicaţia se poate folosi atât în scopuri administrative cât şi în scopuri de *troubleshooting* pentru reţea. Popularitatea acestei aplicaţii este motivată de:

- Un GUI intuitiv şi uşor de utilizat
- Modul facil de prezentare a informaţiei
- Existenţa unor filtre complexe pe baza cărora se poate izola traficul interesant

6.3.1.1 Instalarea Wireshark

Instalarea *Wireshark* se face cu ajutorul utilitarului `apt`.

```
waters@myr:/etc/apt$ sudo apt-get install wireshark
```

Wireshark se poate porni din linie de comandă,

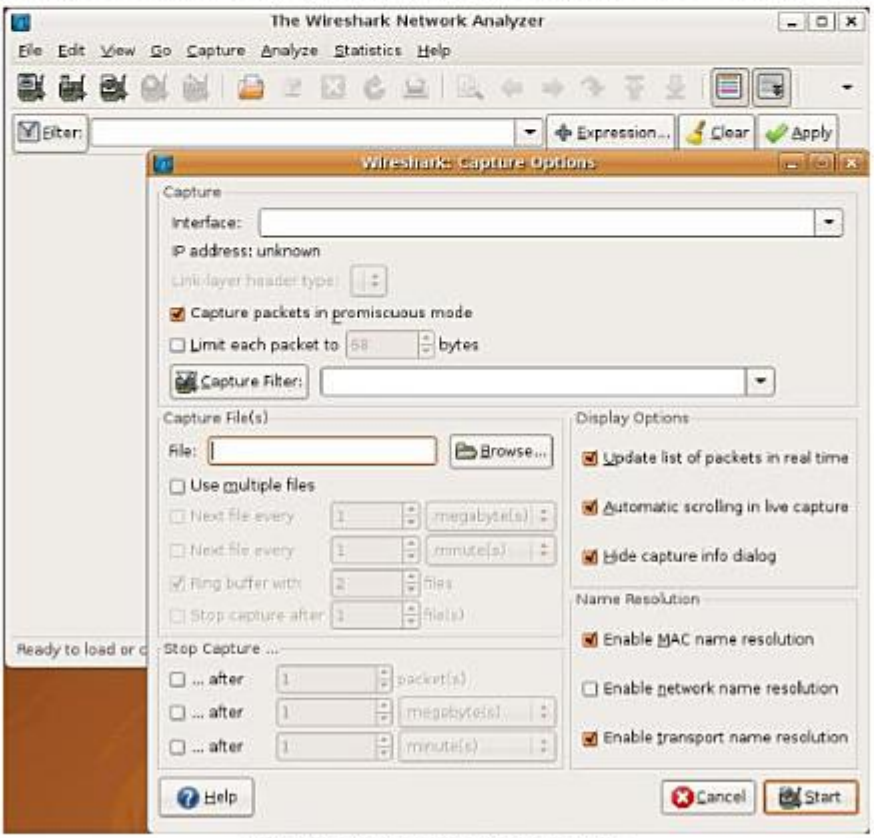
```
waters@myr:/etc/apt$ sudo wireshark &
```

sau, din interfaţa grafică GNOME, din `Applications > Internet > Wireshark`.

***Atenţie!** Pentru a putea captura toate pachete, wireshark trebuie pornit ca root.*

6.3.1.2 Pornirea unei capturi de pachete

Odată pornit, se poate porni captura de pachete pe una din interfeţele active. Selecţia interfeţei şi afişarea opţiunilor de captură se face selectând `Capture > Capture Options`.



6-6: Pornirea capturii de pachete

Material cu drept de autor

Fereastra de opţiuni conţine următoarele setări:

- **Selectarea interfeţei**: se poate alege interfaţa pe care să se facă captura dintr-o listă aflată în partea de sus a ferestrei.
- Opţiunea de a seta interfaţa în **modul promiscuous**: modul acesta impune acceptarea şi captura tuturor pachetelor, nu doar cele destinate staţiei. Această opţiune era folosită în trecut când huburile încă erau utilizate în topologii de reţea. În prezent această opţiune este redundantă pentru reţele cablate pentru că la nivelul doi într-o reţea se va afla mereu un switch care nu permite alt trafic pe un segment, în afară de traficul destinat unei staţii aflate pe acel segment. Totuşi, opţiunea este extrem de utilă în cazul interfeţelor *wireless*.
- **Capture filter**: acesta este câmpul în care se poate introduce un filtru de captură. Implicit, când se porneşte o captură, se vor afişa toate pachete ce ajung la interfaţa de reţea. Dacă se foloseşte un filtru de captură, se poate impune o restricţie asupra pachetelor ce se doresc capturate. Spre exemplu se pot captura doar pachetele cu un anumit IP sursă, sau după protocolul de nivel 4 folosit.
- **Capture file**: oferă posibilitatea de a salva captura într-un fişier pe disc. Acest fişier poate fi încărcat ulterior şi utilizat pentru aplicarea unor filtre de afişare (*display filters*).

Pentru a porni o captură simplă, se apasă butonul de start din fereastra de opţiuni. Odată pornită captura, fereastra principala a aplicaţiei se va împărţi în două componente:

- cadrul de listare a pachetelor capturate
- cadrul de inspectare a pachetelor capturate

Pentru a inspecta un pachet se selectează pachetul din cadrul de listare şi apoi se navighează într-un arbore de antete în cadrul de inspectare.

6.3.1.3 Stream-uri TCP

Una din cele mai importante avantaje *Wireshark*, este că poate afişa *stream*-uri TCP aşa cum sunt văzute de către aplicaţii. În mod implicit *Wireshark* prezintă un *stream* de informaţii pachet cu pachet, iar citirea unui şir de date poate fi destul de dificilă, fiind nevoie să se navigheze prin fiecare pachet. Gruparea în stream TCP reconstruieşte un şir de date, prezentând informaţia într-o structură grupată în ordinea numerelor de secvenţă TCP. Pentru a realiza aceasta grupare în stream, trebuie dat clic dreapta pe un pachet TCP şi selectată opţiunea *Follow TCP Stream*.

6.3.1.4 Filtre

În afara cazului în care se doreşte realizarea unei statistici precum cea descrisă mai sus, foarte rar va interveni situaţia în care se va dori capturarea întregului trafic. Motivul pentru acest lucru este cantitatea mare de trafic neinteresant. Mai ales pe *wireless*, doar protocolul în sine creează cam zece pachete pe secundă prin folosirea de: *beacon*-uri, *ack*-uri şi *broadcast*-uri. Pentru ca este ineficientă parcurgerea unei liste de captură foarte mare pentru a căuta pachetele dorite, *Wireshark* pune la dispoziţie două tipuri de filtre:

- **Filtrul de captură**: se poate aplica din meniul de *Capture Options* şi presupune o captură selectivă bazată pe regulile filtrului
- **Filstrul de afişare**: se aplică peste o captura deja efectuată. Este mai puternic decât filtrul de captură şi este folosit împreuna cu expresii regulate avansate pentru a găsi un anumit tip de pachete într-o captura salvată.

Aplicarea filtrelor de captură se face din meniul *Capture Options*. Se pot aplica mai multe filtre predefinite unei capturi ce urmează să fie făcută. Filtrele de afişare se invocă asemănător, asupra unei capturi deja realizate, din meniul `Analyze > Display Filters`.

Material cu drept de autor

6.3.2 Wireshark – configurări avansate

Wireshark include două mini-limbaje diferite pentru crearea regulilor de filtrare. În cele ce urmează se vor analiza operatorii şi sintaxa limbajul folosit pentru filtrele de afişare.

6.3.2.1 Realizarea filtrelor de afişare

În esenţă, orice câmp ce apare în cadrul de inspectarea a pachetelor poate fi folosit pentru a realiza o regulă. Totalitatea câmpurilor ce pot fi folosite într-o regulă pot fi văzute în meniul `Help > Supported Protocols > Tabul` *„Display Filter Fields”*.

6.3.2.1.1 Operatori

1. **Operatorii de comparaţie** pe care limbajul îi oferă se pot folosi în una din cele două forme posibile: forma literală şi forma matematică:

forma literală	forma matematică
eq	==
ne	!=
gt	>
lt	<
ge	>=
le	<=

2. **Operatorii logici** sunt: *and*, *or*, *xor* şi *not*.

forma literală	forma matematică
and	&&
or	||
xor	^^
not	!

3. **Operatorul boolean** este exprimat în mod implicit prin numele câmpului. De exemplu dacă se foloseşte ca filtru a operatorul **TCP**, se vor selecta doar acele pachete ce au ca protocol de nivel patru TCP. Folosirea operatorului logic **not** în faţa câmpului TCP, are ca rezultat selectarea pachetelor ce nu folosesc TCP.
4. **Adresele IP:** sunt suportate protocoalele Ipv4 şi IPv6. Este suportată şi notaţia CIDR: 192.168.1.0/24.

6.3.2.1.2 Câmpurile de filtrare

Regulile de filtrare au două componente: operatorii şi câmpurile. Acestea din urmă sunt specificate precum într-un limbaj orientat pe obiecte. Dacă se doreşte, spre exemplu, accesare câmpului <IP sursă> din headerul IP, formularea se va face astfel: **ip.src**. Sintaxa generala se deduce, astfel, a fi:

```
protocol.camp_antent
```

În situaţia în care există un câmp inclus într-un alt câmp, acestea se pot înlănţui în aceeaşi maniera: **telnet.auth_mod.name;** acest câmp identifică pachetele ce conţin numele de autentificare pentru protocolul *telnet*.

Material cu drept de autor

6.3.2.1.3 Operatori pe string-uri

Wireshark pune la dispoziţie şi metode avansate de selecţie **în interiorul** unui câmp. Pentru a explica mai uşor sintaxa, aceasta se va urmări pe exemple practice de mai jos:

- `eth.src[0:2]` – acesta expresie va selecta primii 2 octeţi din adresa MAC sursă. Formatul `[x:y]` va referii y octeţi din adresa MAC, începând cu octetul x.
- `eth.src[3-4]` – expresia va selecta octeţii 4 şi 5 din adresa MAC sursă. Formatul `[x-y]` va referii octeţii cu intexul 3 şi 4 din adresa MAC
- `eth.src[1]` - expresia va selecta octetul 2 din adresa MAC.

6.3.2.1.4 Aplicarea unui filtru de afişare

Regula de filtrare se introduce în câmpul poziţionat deasupra cadrului de afişare a pachetelor:

6-7: Aplicarea unui filtru de afişare

6.3.2.1.5 Exemple de reguli de filtrare

1. Selectarea tuturor pachetelor care au adresa IP din clasa 10.0.0.0/24 şi portul UDP destinaţie 53

```
ip.addr == 10.0.0.0/24 && udp.srcport == 53
```

2. Selectarea tuturor pachetelor cu adresa IP sursă 10.0.0.5 care au bitul de TCP FIN setat.

```
ip.src == 10.0.0.5 and tcp.flags.fin
```

3. Selectarea tuturor pachetelor care nu au adresa IP 10.0.0.5, nici ca IP sursă, nici ca IP destinaţie

```
!(ip.addr == 10.0.0.5)
```

Atenţie! *Expresia ip.addr != 10.0.0.5 va evalua întotdeauna* ***true****, pentru că va testa ca cel puţin unul din câmpurile IP să fie diferit de 10.0.0.5.*

4. Selectarea tuturor pachetelor Ipv4 care au câmpul TTL mai mare sau egal cu 1 şi care au primii 3 ovteţi ai MAC-ul de la un vendor specific.

```
ip.version eq 4 and ip.ttl >=2 and eth.addr[0-2] == 00:1A:5E
```

Bineînţeles că regulile se pot complica, depinzând de scenariu, însă până în acest punct s-au descris elementele de bază a limbajului filtrelor de afişare.

6.3.2.1.6 Construirea expresiilor regulate în GUI

Wireshark oferă suport grafic pentru crearea regulilor cu majoritatea câmpurilor din diferite protocole, indicând chiar şi valori posibile pentru unele dintre ele. Interfaţa de construire a filtrelor de *display* se accesează cu ajutorul butonului *„Expression…"*

6.3.3 Snort – captură de pachete în linie de comandă. IDS/IPS.

Snort este un *software open-source* folosit pentru a filtra trafic şi a detecta tipare de trafic ce pot reprezenta o ameninţare pentru o reţea de calculatoare. Deşi cunoscut ca unul din cele mai populare **IDS** –uri (*Intrusion detection System*), snort a fost la origini un *sniffer* de pachete, nu foarte diferit de `tcpdump`. De fapt, atât tcpdump cât şi snort folosesc biblioteca open-source **libpcap**, care permite analiza pachetelor pe un sistem Linux.

Aplicaţia are trei moduri de funcţionare:

Material cu drept de autor

- Modul simplu de captură de pachete – în acest mod `snort` capturează traficul definit ca trafic interesant
- Modul de captură de pachete cu jurnalizare – `snort` poate să stocheze captura într-un fişier pe disc în diferite formate: format *syslog*, text, binar.
- Modul IDS/IPS – în acest mod, `snort` permite definirea de reguli de identificare a unui tipar de trafic şi generare de alerte, invocarea de alte programe, etc.

6.3.3.1 Instalarea snort

Programul se instalează din apt:

```
root@myr:/home/rl#  apt-get install snort
```

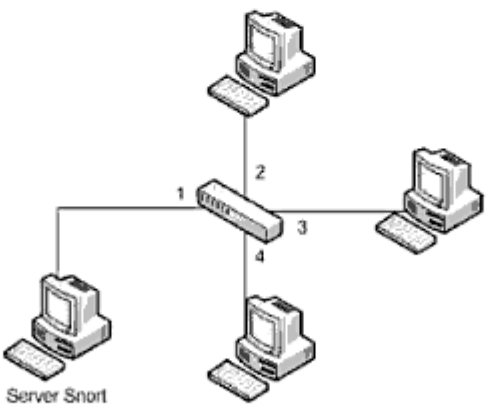
În timpul instalării, aplicaţia va configura într-un mod interactiv interfaţa activă pe care `snort` va asculta implicit şi subnetul de reţea din care va accepta pachete.

După ce a fost instalat, se poate verifica existenţa utilitarului în calea implicită, rulând comanda `snort`. Aplicată fără niciun parametru, comanda va afişă toate opţiunile `snort` alături de un mesaj explicativ.

```
root@myr:/home/rl#  snort
[...]
Uh, you need to tell me to do something...
```

6.3.3.2 Modurile de funcţionare snort

S-a menţionat anterior faptul ca `snort` a fost la început doar un utilitar folosit pentru captură de pachete în linie de comandă. În timp el a evoluat spre a îndeplini mai multe sarcini, însă fără a-şi pierde funcţionalitatea de bază: *sniffer*. Dar, revenind la teoria prezentată anterior în carţe, folosirea unui switch într-o reţea *Ethernet* asigură faptul că fiecare staţie de pe segment, va primi numai traficul destinat acesteia. Cum poate deci un administrator în reţelele din prezent, să folosească un *sniffer*? Cea mai folosită soluţie în implementări este tehnica de **port mirroring**. Aceasta presupune configurarea unui switch, pentru ca traficul care traversează anumite porturi să fie copiat pe un port special configurat de administrator.



6-8: Port mirroring

Folosind port mirroring în configuraţia de mai sus, se poate configura switchul astfel încât traficul de pe porturile 2, 3 şi 4 să fie copiat pe portul 1 pe care se află serverul ce rulează `snort`.

Se vor prezenta pe scurt în continuare cele trei moduri în care `snort` operează:

Material cu drept de autor

6.3.3.2.1 Modul simplu de captură

`snort` se rulează după cum s-a prezentat mai sus, în linie de comandă. Sintaxa pe care utilitarul o foloseşte este următoarea:

```
snort [options] expression
```

Expresia pe care snort o primeşte ca argument este folosită pentru a defini filtre în linia de comandă. Există mai multe primitive şi cuvinte cheie care se pot folosi, dar deoarece abordarea prezentării acestui utilitar va fi una bazată mai mult pe exemple şi funcţionalitate, acestea nu vor fi enumerate aici. Pentru o lista completă a primitivelor şi expresiilor se poate consulta pagina `man` a aplicaţiei.

Se va crea următorul scenariu:

- Pe o staţie din subnetul 192.168.1.0/24 se va rula `snort` în modul simplu de captură şi în acelaşi timp un server FTP (*proftpd*)
- De pe o staţie din alt subnet (192.168.0.0/24), se va iniţia o conexiune FTP pe portul 21 către serverul pe care `snort` ascultă.
- Scopul va fi capturarea *user*-ului şi parolei FTP cu ajutorul `snort`, trimise în text clar pe reţea

Mai întâi se va porni `snort` folosind o expresie regulată ce va captura doar traficul cu IP destinaţie 192.168.1.2 (adică staţia locală) şi cu portul destinaţie 21 (FTP).

```
root@myr:/home/rl#  snort  -vde host 192.168.1.2 and port 21
```

Parametrii `-v -d` şi `-e` aplică următoarele opţiuni:

- `-v`: printează captura la ieşirea standard
- `-d`: afişarea informaţiei de la nivelul aplicaţie
- `-e`: afişarea *header*-ului de nivel 2 într-un *log* sau la ieşirea standard

În timp ce `snort` rulează, se va porni şi un server FTP pe aceeaşi staţie:

```
root@myr:/home/rl#  /etc/init.d/proftpd start
```

De pe staţia cu IP-ul 192.168.0.254, se va face o conexiune către serverul FTP:

```
root@myr:/home/rl#  ftp 192.168.1.2
Connected to 192.168.1.2.
220 ProFTPD 1.3.1 Server (Debian) [192.168.1.2]
Name (192.168.1.2:rl): rl
331 Password required for rl
Password:
230 User rl logged in
[...]
```

După cum se poate observa din rezultatul de mai jos, `snort` a capturat informaţia din pachet de la nivel 2 la nivel 7.

```
=+=+=+
```

Problema folosirii afişării la ieşirea standard este faptul ca `snort` poate captura mult mai repede decât poate afişa pe ecran. De aceea la un trafic susţinut, `snort` va începe să sară peste unele pachete pentru a putea afişa informaţia pe monitor. De aceea, de obicei, `snort` se rulează cu opţiunea `-l`, care permite salvarea capturii în jurnale.

6.3.3.2.2 Modul de captura de pachete cu jurnalizare

Pentru simplitate, se va folosi acelaşi scenariu ca şi în subcapitolul anterior. La instalare, snort îşi creează un director special folosit pentru jurnale şi alerte: `/var/log/snort`. Există două moduri diferite de jurnalizare:

- Modul *clear text*: este mai încet, însă se poate citi uşor informaţia din *log*-uri.
- Modul binar: este cel mai rapid mod de jurnalizare, însă e nevoie de utilitare speciale pentru a citi logurile. De asemenea formatul binar este compatibil cu `tcpdump` şi oferă şi avantajul posibilităţii aplicării a mai multe reguli `snort` asupra capturii, pentru a analiza pachetele pentru un posibil atac.

Modul de logare în format clear-text

În continuare se va rula din nou `snort`, însa de acestă dată, cu jurnalizare:

```
root@myr:/home/rl#  snort  -vde host 192.168.1.2 and port 21 -l /var/log/snort/ -K ascii
```

Opţiunea `-K` este folosită pentru a preciza tipul jurnalizării.

După ce se parcurg aceeaşi paşi ca mai sus: pornirea serverului FTP şi *log*-area, se va observa că în directorul `/var/log/snort` s-a creat un director al cărui nume, este **IP-ul sursă** al pachetelor de conexiune FTP: 192.168.0.254..

```
root@myr:/var/log/snort#  ls
192.168.0.254
root@MZ:/var/log/snort#  file 192.168.0.254/
192.168.0.254/: setgid directory
```

Doar în cazul folosirii jurnalizării în format ASCII se creează câte un director pentru fiecare IP sursă care deschide o conexiune. În acest director se află un fişier *clear-text,* în care se află stocată captura.

```
root@myr:/var/log/snort#  ls 192.168.0.254/
TCP:56443-21
root@myr:/var/log/snort#  file 192.168.0.254/TCP\:56443-21
TCP:56443-21: ASCII text
```

Modul de logare în format binar

De această dată, parametrul opţiunii `-K` este `pcap`, pentru jurnalizarea în format binar compatibil cu `tcpdump`.

```
root@myr:/home/rl#  snort  -vde host 192.168.1.2 and port 21 -l /var/log/snort/ -K pcap
```

Acest tip de jurnalizare este mult mai rapid şi spre deosebire de cel modul *clear-text,* captura este păstrată într-un singur fişier:

```
root@myr:/var/log/snort#  ls
snort.log.1220806500
root@myr:/var/log/snort#  file snort.log.12/0806500
snort.log.1220806500: tcpdump capture file (little-endian) - version 2.4 (Ethernet,
capture length 1514)
```

Snort oferă bineînţeles şi un mod facil de a citi informaţia dintr-o captură în format binar, folosind opţiunea `-r`:

```
root@myr:/var/log/snort#  snort -dev -r snort.log.1220806500
[...]
```

Material cu drept de autor

Deasemenea, formatul `pcap` este folosit şi de aplicaţii cu interfaţa GUI, precum *Wireshark*. Deci captura poate fi încărcată şi vizualizată şi în mod grafic.

6.3.3.3 Modul IDS/IPS

Deşi modul de logare al utilitarului este foarte util, `snort` a devenit faimos pentru capabilităţile sale de **IDS**.

> *Un **IDS** este un sistem software sau hardware folosit pentru detectarea unei încercări de acces, manipulare, sau atac asupra unui sistem.*

Acţiunea de bază pe care orice IDS trebuie o întreprinde la detectarea unui trafic ce ridică suspiciuni, este provocarea unei alarme în sistem. În continuare se va analiza modul în care poate fi folosit `snort` pentru a îndeplini această funcţie.

Înainte de a putea porni `snort` în modul IDS, trebuie modificat fişierul principal de configurare: `/etc/snort/snort.conf`. Trebuie specificat care este reţeaua pe care `snort` trebuie sa o protejeze şi care este reţeaua de la care se pot aştepta atacuri. Pentru acest lucru, snort defineşte două variabile: `$HOME_NET` şi `$EXTERNAL_NET`. Aceste sunt implicit setate pe any în fişierul de configurare. Doar `$HOME_NET` va trebui schimbată la adresa de reţea a LAN-ului.

```
root@myr:/home/rl#  cat /etc/snort/snort.conf
var HOME_NET 192.168.1.0/24

#  Set up the external network addresses as well.  A good start may be "any"
var EXTERNAL_NET any
```

Din configuraţia de mai sus, `snort` va considera ca reţeaua ce trebuie protejată este reţeaua 192.168.1.0/24, iar orice alta reţea (`var EXTERNAL_NET any`) este văzută ca o posibilă ameninţare.

Pentru a putea demonstra modul în care `snort` acţionează se va presupune următorul scenariu:

- De pe staţia cu IP-ul 192.168.0.254 de pe care s-a iniţiat conexiunea FTP în exemplul anterior, se va porni acum o scanare `nmap` cu *fingerprint*-ing.
- `snort` va detecta această încercare de scanare şi va crea un fişier de alarmă în `/var/log/snort/` care se va putea interpreta cu utilitare speciale.

Mai întâi se va reporni snort pentru a asigura reinterpretarea fişierului de configurare:

```
root@myr:/home/rl#  /etc/init.d/snort restart
 * Stopping Network Intrusion Detection System  snort                    [ OK ]
 * Starting Network Intrusion Detection System  snort                    [ OK ]
```

Rularea se poate verifica prin listarea procesului în sistem:

```
root@myr:/var/log/snort#  ps -ef | grep snort
snort     18215     1  8 21:27 ?        00:00:06 /usr/sbin/snort -m 027 -D -d -l
/var/log/snort -u snort -g snort -c /etc/snort/snort.conf -S HOME_NET=[192.168.0.0/16] -i eth2
```

Acum că `snort` rulează, nu trebuie decât să pornim scanarea de porturi. Se va face o scanare cu pachete TCP SYN şi un *fingerprint*.

```
root@myr:/home/rl#  nmap -sS -O 192.168.1.2
```

După ce scanarea s-a terminat se poate verifica `/var/log/snort/`.

```
root@myr:/var/log/snort#  ls -l
total 44
drwx--S--- 2 root  adm  4096 2008-09-07 19:46 192.168.0.254
-rw-r----- 1 snort adm  5875 2008-09-07 21:33 alert
-rw------- 1 root  adm 15369 2008-09-07 21:33 snort.log.1220806500
-rw------- 1 root  adm 12212 2008-09-07 21:33 snort.log.1220812428
-rw-r----- 1 snort adm  1871 2008-09-07 21:33 tcpdump.log.1220812063
```

Material cu drept de autor

S-a generat un fișier de alertă de aproximativ 6 KB. Pentru a putea interpreta acest fișier se pot folosi destul de multe utilitare, atât în linie de comandă, cât și în mediu grafic. Unul din cele mai cunscute este `snortsnarf`. Acest utilitar poate interpreta un fișier de alarmă generat de `snort` și îl poate afișa în format HTML cu *link*-uri utile către diferite *site*-uri ce pot spune mai multe despre tipul de atac interceptat.

6.4 *Securitate și monitorizare în* Windows Server 2008

Protejarea rețelei interne împotriva atacurilor din exterior este o responsabilitate extrem de importantă, mai ales când rețeaua locală trebuie conectată la o rețea publică, cu un acces mult mai puțin restricționat, ca Internetul. *Windows Server 2008* implementează o serie de strategii pentru asigurarea securității de sistem, printre care *Windows Firewall* și IPSec (protocolul *IP Security*).

6.4.1 *Windows Firewall* with Advanced Security

Un *firewall* este definit ca o componentă *software* sau *hardware*, interpusă între sistem (rețea) și Internet (de cele mai multe ori) cu rolul de a proteja împotriva acceselor neautorizate din exterior, analizând datele care circulă în ambele sensuri și luând decizii de filtrare (blocare) atunci când este cazul. Deciziile de filtrare a pachetelor se iau conform cu regulile definite în *firewall*. Dacă pachetele care sosesc din exterior nu se conformează niciunei reguli din *firewall*, acestea sunt filtrate, în mod implicit. De asemenea, *firewall*-ul poate fi setat să verifice și pachetele care părăsesc sistemul sau rețeaua proprie prin instituirea unor reguli ce controlează accesul anumitor aplicații spre Internet.

Windows Firewall este unul de tip *stateful*, adică urmărește starea conexiunilor de rețea ale unui sistem, examinând atât traficul direcționat spre rețea cât și cel spre exterior. Pentru traficul dinspre exterior, comportamentul implicit al său este de a bloca orice pachet care soseşte nesolicitat, adică nu reprezintă răspunsul la o cerere sau nu face parte din traficul unei conexiuni inițiate de calculatorul propriu. Totuși, când este necesar, pot fi configurate excepții pentru a permite anumitor tipuri de trafic să fie recepționate de către anumite aplicații, pe anumite porturi.

Comportamentul implicit pentru traficul originar din sistemul propriu este unul permisiv, nefiind filtrate niciun fel de cereri. Totuși, pot fi implementate reguli care să limiteze accesul anumitor aplicații la Internet

6.4.1.1 Configurarea *Windows Firewall*

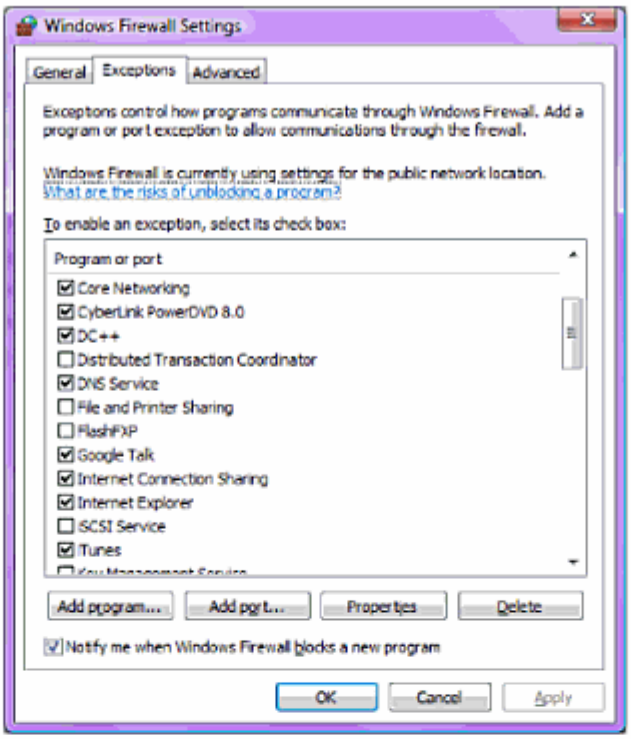
Pentru configurarea *Windows Firewall* pe un sistem *Windows Server 2008* există două moduri: primul dintre ele îl reprezintă fereastra de dialog *Windows Firewall Settings*, disponibilă și pentru Windows XP și accesibilă direct din *Control Panel*; cea de-a doua este interfața de administrare pentru *Windows Firewall with Advanced Security*, accesibilă de la *Administrative Tools* sau din *Server Manager*.

Pentru a deschide fereastra de dialog *Windows Firewall Settings*, se poate alege opțiunea *Allow a Program Through the Windows Firewall* din *Control Panel*, sub categoria *Security* sau direct accesând *Windows Firewall*, tot din Control Panel, ca și în Windows XP (în funcție de modul de vizualizare activ).

Setările de bază ale lui *Windows Firewall* sunt separate în trei categorii:

- **General**: În acest tab se găsesc cele mai simple opțiuni, de pornire și oprire a *firewall*-ului. De asemenea, există și posibilitatea de blocare a tuturor conexiunilor inițiate din exterior (*Block all incoming connections*) pentru a activa rapid protecția totală în rețele publice, nesecurizate. Această opțiune ignoră toate excepțiile active la acel moment.

Material cu drept de autor



6-1: Aplicaţiile permise prin *Windows Firewall*

- **Exceptions**: Lista cuprinde aplicaţiile detectate în sistem iar bifarea lor are ca efect activarea permisiunii aplicaţiilor de a deschide porturi. Pot fi adăugate noi programe (executabile, de fapt) la lista de excepţii sau pot fi adăugate separat şi porturi.
- **Advanced**: Aici pot fi selectate conexiunile de reţea pe care *Windows Firewall* să le monitorizeze. De asemenea, de aici pot fi restaurate setările implicite legate de funcţionarea *firewall*-ului şi de excepţiile sale.

După cum se observă, fereastra de dialog descrisă mai sus nu oferă o multitudine de opţiuni legate de configurarea *Windows Firewall* şi nu poate fi considerată efectiv o unealtă administrativă deoarece oferă un grad extrem de scăzut de flexibilitate.

Dacă se doreşte modificarea unor setări mai avansate (reguli, în speţă, care, la rândul lor, formează politicile de securitate) trebuie utilizată interfaţa *Windows Firewall with Advanced Security*, accesibilă din *Server Manager* sau de la `Start > Administrative Tools`. Aceasta oferă opţiunea de a gestiona regulile de intrare şi ieşire a pachetelor şi de a crea reguli de securitate pentru conexiuni care pot restricţiona conectarea la un server pe baza unor informaţii de autentificare mai complexe, cum ar fi apartenenţa la un domeniu.

Interfaţa este împărţită în trei panouri:

- în stânga, un panou ce afişează diferitele elemente de configurare, cum sunt regulile sau opţiunile de monitorizare;
- în partea dreaptă panoul de acţiuni, comun celor mai multe interfeţe de administrare din *Server Manager*;
- în partea centrală panoul de detalii, al cărui conţinut se modifică dinamic, în funcţie de selecţiile din primul panou.

Material cu drept de autor

6-2: Interfața de administrare *Windows Firewall* with Advanced Security

În mod implicit, panoul central, de detalii, afișează (atunci când în stânga este selectată rădăcina *firewall*-ului: *Windows Firewall with Advanced Security on Local Computer*) o listă cu trei profiluri: **Domain**, **Public** și **Private**. Aceste profiluri se află în legătură cu tipurile de conexiuni afișate și de interfața *Network and Sharing Center* (prezentată în secțiunea 3.4.1) și conțin setări diferite în funcție de riscurile pe care diferite tipuri de conexiuni le prezintă. Ele au relevanță în contextul lui *Windows Firewall* după cum urmează:

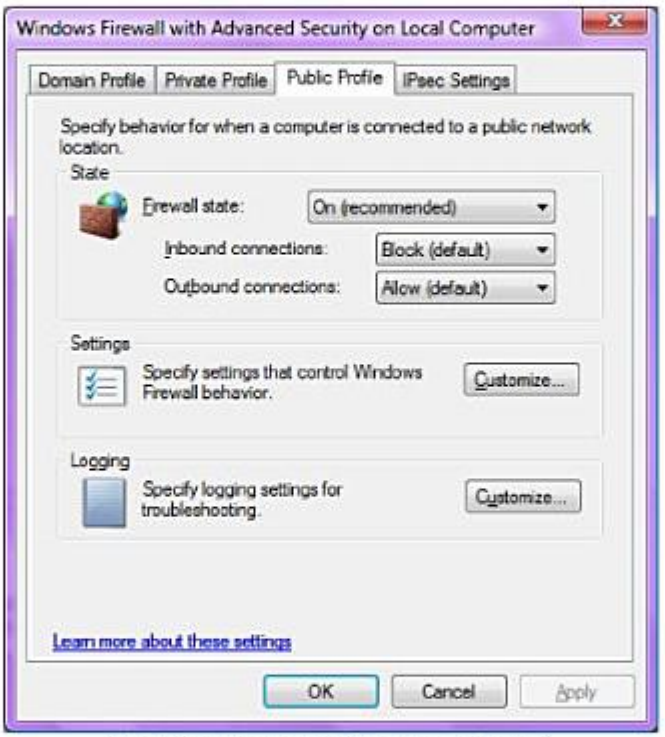
- **Domain**: Calculatoarele ce rulează Windows *Vista* sau *Windows Server 2008* pot detecta dacă se poate realiza apartenența la un domeniu într-o anumită rețea la care sunt conectate. Acest profil necesită ca toate calculatoarele să fie membre ale unui domeniu pentru a putea accesa *controller*-ul de domeniu.
- **Public**: Profilul *Public* este folosit de către *firewall* pentru a proteja sistemul când acesta este conectat la o rețea publică, spre exemplu una *wireless*. Practic, pentru *Windows Server 2008*, o rețea publică reprezintă orice rețea care nu se află în interiorul perimetrului rețelei delimitate de *firewall*-ul acesteia.
- **Private**: Profilul comunică *firewall*-ului modul în care să protejeze sistemul în momentul în care acesta este membru al unei conexiuni private, adică aparține unei rețele protejate de un *firewall hardware*.

Pentru că fiecare dintre cele trei profiluri poate stoca setări distincte cu privire la regulile *firewall*-ului, acestea oferă un grad sporit de flexibilitate în optimizarea nivelului de securitate oferit de *firewall* pentru diferite tipuri de conexiuni. Spre exemplu, conexiunile spre rețele

Material cu drept de autor

publice vor folosi profilul *Public* care va impune un set de reguli mai restrictive, în timp ce conectarea la reţelele locale izolate, securizate şi/sau aflate sub propria administrare poate necesita un set mai permisiv de reguli, cum ar fi partajarea fişierelor şi a imprimantelor în reţea, reguli configurate în cadrul profilului *Private*.

Modificarea tipului unei conexiuni se poate face manual (mai puţin pentru tipul *Domain*, care are cerinţe suplimentare). Windows aplică, însă, şi în mod automat aceste profiluri în funcţie de tipul de trafic inspectat pe interfaţa conectată. Astfel că, în cazul apartenenţei la un domeniu, se aplică întâi profilul *Domain*, ajungându-se ulterior la opţiunea de a aplica unul dintre celelalte două profiluri. Pentru situaţia de mai sus, peste profilul *Domain* se aplică automat profilul *Private* deoarece se consideră că zona delimitată de staţiile dintr-o reţea, membre ale unui domeniu, reprezintă deja o zonă de un anumit grad de securitate şi siguranţă. Dacă interfaţa nu este autentificată la un *controller* de domeniu (deci conexiunea sa nu este membră a unui domeniu) atunci se aplică automat profilul *Public*.



6-3: Interfaţa proprietăţilor unui profil

Pentru a accesa setările profilurilor implicite, se poate face clic pe *link*-ul *Windows Firewall Properties* din panoul de detalii, la baza regiunii *Overview*, în care sunt listate aceste profiluri (figura 6-3).

Se observă că toate cele trei profiluri se configurează similar, specificându-se starea *firewall*-ului şi modul de tratare a conexiunilor în funcţie de sensul în care au fost iniţiate. La apăsarea butonului *Customize* sunt disponibile opţiuni suplimentare ce adresează notificarea utilizatorului în momentul în care *firewall*-ul blochează o aplicaţie şi comportamentul permis de *firewall* în cazul în care sistemul încearcă să răspundă prin *unicast* în urma unui trafic de *broadcast* sau *multicast* din reţea.

Material cu drept de autor

6.4.1.2 Modificarea regulilor implicite

Luând în considerare setările anterioare, disponibile la nivel de profil, este evident faptul că diferențierea cea mai drastică şi totodată cea mai granulară dintre profiluri se reduce la implementarea regulilor. Practic, la nivel de trafic, regulile sunt cele care dictează comportamentul lui *Windows Firewall*. Acestea se împart în trei categorii: **inbound rules** (pentru traficul care „intră" printr-o conexiune), **outbound rules** (pentru traficul adresat spre exterior) şi **connection security rules**.

Regulile de tip *inbound* se referă, de fapt, la „deblocarea" traficului venit din exterior. După cum s-a menționat şi în secțiunea anterioară, pentru toate cele trei tipuri de profiluri (*Domain*, *Private* şi *Public*), comportamentul implicit al *firewall*-ului pentru conexiunile inițiate din exterior este de a le bloca. În contrast, comportamentul implicit pentru conexiunile inițiate de maşina pe care rulează *firewall*-ul este permiterea tuturor acestora prin *firewall*, astfel că regulile de tip *outbound* adresează care dintre aceste conexiuni vor fi, de fapt, blocate.

Atât pentru regulile de tip *inbound* cât şi pentru cele *outbound*, categoriile de reguli pe care *Windows Firewall* permite să fie create sunt în număr de trei, cu posibilitatea de a crea şi reguli *Custom*:

- **Program**: Decizia se ia în funcție de programul care inițiază o conexiune (regulă *outbound*) sau care dorește deschiderea unui port (regulă *inbound*). Informația care identifică aplicația cuprinde doar calea spre executabilul său.
- **Port**: Astfel de reguli au în vedere conexiunile pe baza numerelor de port (unul sau un interval) pe care acestea le utilizează. Tot aici se poate specifica şi pentru ce protocol de nivel transport (TCP sau UDP) se aplică regula.
- **Predefined**: Aceste reguli generalizează anumite programe sau servicii ale sistemului simplificând detaliile funcționării lor. Spre exemplu, o astfel de regulă poate fi instituită pentru a controla accesul la partajarea fişierelor sau imprimantelor sau pentru a permite funcționarea protocolului *Remote Desktop* (pentru administrarea de la distanță).

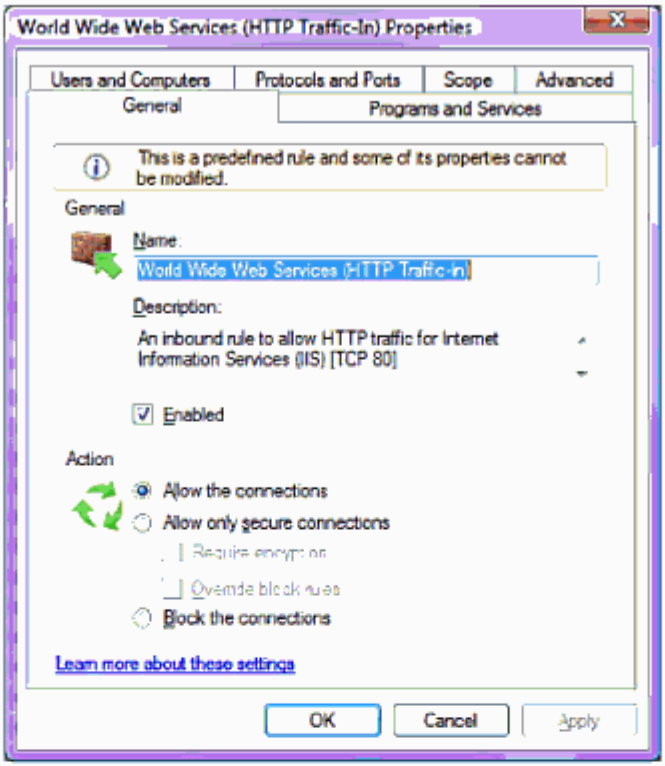
Afişarea regulilor definite implicit în *Windows Firewall* se face printr-un simplu clic pe categoria dorită în panoul din stânga: *Inbound Rules* sau *Outbound Rules*. Deoarece lista poate deveni destul de cuprinzătoare, mai ales după definirea regulilor proprii şi pe servere cu o multitudine de aplicații şi servicii instalate, acesteia i se pot aplica filtre din panoul de acțiuni[1]. Aplicarea filtrelor poate adresa profilul de care acestea aparțin, starea lor (active sau nu) şi grupul din care fac parte (adică aplicația sau serviciul de care aparțin). Filtrele pot fi aplicate secvențial. Pentru ştergerea tuturor filtrelor active se face clic pe *link*-ul *Clear all filters*.

Pentru a vizualiza proprietățile unei reguli (indiferent dacă este de tip *inbound* sau *outbound*), se poate face dublu clic pe regulă din panoul central (indiferent de prezența filtrelor).

***Atenție**, nu se pot modifica toate proprietățile regulilor predefinite; în cele mai multe dintre cazuri, executabilul asociat regulii nu poate fi schimbat, la fel ca şi porturile şi setul de protocoale incluse în regulă. Aceste restricții sunt impuse deoarece regulile implicite adresează funcționarea serviciilor Windows care folosesc porturi şi protocoale standard pentru a comunica.*

[1] Este de la sine înțeles că filtrele afectează doar afişarea anumitor reguli; ele nu afectează în niciun fel funcționarea lor. Pentru aceasta se foloseşte proprietatea Enabled disponibilă pentru fiecare regulă.

Material cu drept de autor



6-4: Proprietăţile unei reguli *inbound*

Fereastra de dialog a proprietăţilor unei reguli conţine următoarele secţiuni:

- **General**: Cuprinde numele şi o descriere textuală a regulii, oferă posibilitatea de a activa sau dezactiva regula şi tipul acţiunii descrise de regulă: permiterea conexiunii, blocarea ei sau permiterea în condiţii securizate[1] (*IPSec*). Dacă regula cere ca o conexiune permisă să fie criptată, pentru cele necriptate se vor aplica alte reguli dacă există sau se vor conforma comportamentului implicit descris în profilul conexiunii. Opţiunea *Override block rules* specifică faptul că această regulă va suprascrie orice alte reguli care ar putea bloca conexiunea în cauză. Opţiunea este necesară pentru a permite o conexiune pentru că, în mod normal, regulile de blocare au prioritate sporită faţă de cele permisive.
- **Programs and Services**: Specifică executabilul sau serviciul de sistem pentru care se va aplica regula. Orice program şi serviciu poate fi adăugat atâta timp cât rulează prin propriul său executabil. Atenţie la adăugarea *container*-elor de servicii sau a programelor care găzduiesc executabile, ca *svchost.exe*, *dllhost,exe*, *inetinfo.exe* pentru că pot reprezenta riscuri de securitate. Regulile care se aplică pentru un anumit program sau serviciu pot fi folosite şi pentru a permite unor aplicaţii să accepte conexiuni din Internet, atâta timp cât acestea folosesc *Windows Sockets* (*winsock*) pentru a deschide propriile porturi.
- **Users and computers:** Permite specificarea căror calculatoare sau utilizatori (sau grupuri de utilizatori) au dreptul de a se conecta la calculatorul local în contextul serviciului sau programului căruia regula îi este asociată. Opţiunile legate de utilizatori şi calculatoare pot fi utilizate doar dacă s-a specificat ca acţiune a regulii permiterea conexiunilor dacă acestea sunt securizate. De asemenea, configuraţia este valabilă doar pentru reguli de tip *inbound*, iar utilizatorii sau calculatoarele ce se pot autentifica trebuie să fie accesibile prin *Active Directory Domain Services*.

[1] Pentru aceasta trebuie să existe şi o regulă corespunzătoare de tipul Connection Security Rule.

- **Protocols and ports**: Regula poat fi particularizată în continuare specificând porturile şi protocoalele (TCP, UDP, GRE, IPv6, L2TP, etc.) necesare pentru o conexiune prin această regulă. Pentru ICMP sunt disponibile opţiuni suplimentare, în funcţie de codurile mesajelor. De asemenea, filtrarea pe bază de port se poate face atât pentru porturile sursă cât şi pentru cele destinaţie (proprii, în cazul de faţă).
- **Scope**: Sunt permise specificarea unor adrese IP, intervale de adrese IP sau chiar subreţele întregi de la care sunt acceptate conexiunile. Aceiaşi parametri pot fi configuraţi şi pentru maşina locală, caz în care regula se va aplica tuturor conexiunilor dintre adresele locale şi adresele de la distanţă care îndeplinesc ambele criterii configurate.
- **Advanced**: Se poate specifica profilul pentru care regula este activă (toate sau numai unul dintre cele trei) şi conexiunea de reţea pentru care regula se aplică.

6.4.1.3 Adăugarea de noi reguli

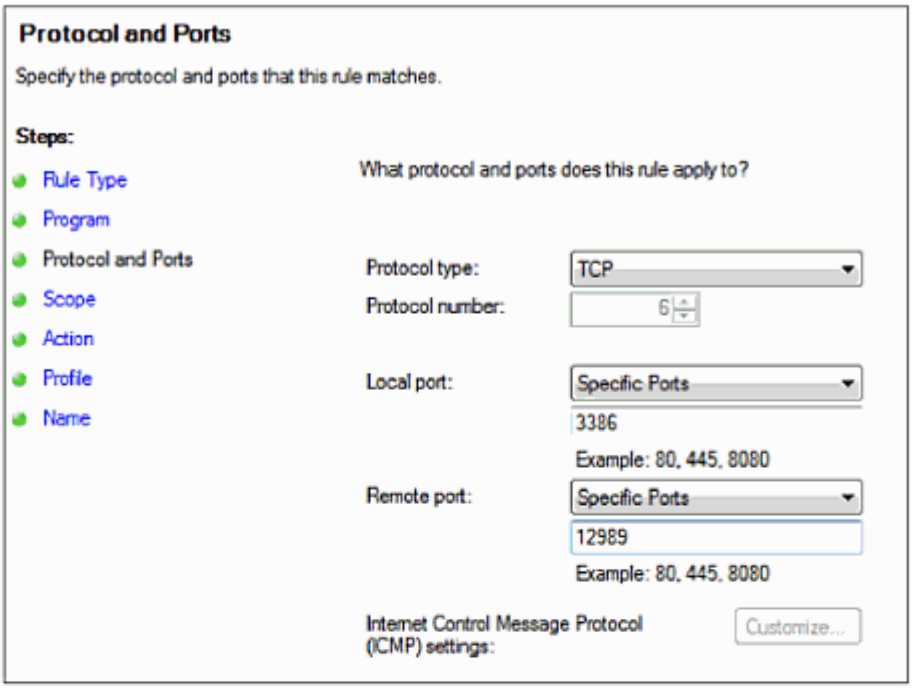
Windows Firewall permite crearea de noi reguli pentru a suplimenta cele implicite, particularizate pentru necesităţile fiecărui sistem sau reţea. Pentru a crea o nouă regulă, se selectează categoria de *Inbound* sau *Outbound* şi se face clic pe *New Rule* în panoul de acţiuni.

1. În prima etapă se selectează tipul regulii: **Program**, **Port**, **Predefined** sau **Custom**, după cum au fost prezentate în secţiunea anterioară. Pentru a avea acces la toate opţiunile, pentru exemplul de faţă se va considera că se creează o regulă de tip **Custom**.
2. Următoarea pagină oferă trei opţiuni:
   - **All programs**: Regula va fi aplicată tuturor programelor ale căror conexiuni se potrivesc cu setările regulii.
   - **This program path**: Regula se va aplica doar conexiunilor iniţiate din sau spre un anumit program selectabil prin executabilul său.
   - **Services**: Permite selectarea unui anumit serviciu din lista de servicii instalate în sistem, deoarece majoritatea rulează "găzduite" în interiorul altor executabile, ca `services.exe`[1] sau `lsass.exe`[2].
3. În continuare (figura 6-5) se poate alege protocolul şi, eventual, porturile pentru care regula va fi aplicată. Dacă la pasul anterior s-a selectat o aplicaţie, nu e necesară selectarea protocolului deoarece Windows îl va identifica din interfaţa *winsock*.
4. În continuare se pot specifica adresele IP, atât locale cât şi de la distanţă pe a căror conexiuni se va aplica regula. Pot fi definite adrese, intervaluri de adrese sau subreţele. De asemenea, se pot alege aici şi tipurile conexiunilor de tip reţea pe care va fi aplicată regula.
5. Acţiunea ce poate fi aplicată în momentul în care regula *firewall*-ului intră în funcţiune, poate să permită realizarea conexiunii în orice situaţie (deci crearea unei reguli de tip **Allow**), doar în cazul în care conexiune este securizată (mai multe detalii în secţiunea 6.4.1.2) sau poate bloca realizarea conexiunii (crearea unei reguli de tip **Deny**).
6. În următoarea secţiune se pot bifa profilurile pentru care regula să se aplice: *Private*, *Public* sau *Domain*.
7. În fine, în ultima etapă se dă un nume regulii şi, eventual o descriere care rezumă parametrii şi acţiunile sale (preferabil şi scopul pentru care a fost creată regula).
8. Butonul *Finish* creează regula, ce poate fi editată ulterior ca şi orice altă regulă implicită (conform secţiunii 6.4.1.2).

[1] Utilitar responsabil cu pornirea şi oprirea serviciilor, pornirea automată a lor la iniţializarea sistemului si oprirea lor la închiderea sa.

[2] Utilitar cu multiple responsabilităţi de securitate, incluzând autentificarea utilizatorilor. Ţintă pentru numeroşi viruşi.

Material cu drept de autor



6-5: Specificarea porturilor şi protocolului pentru regulă

6.4.1.4 Reguli de securizare a conexiunilor

Regulile de tip *inbound* şi *outbound* controlează strict fluxurile de date dintr-un calculator. *Windows Firewall* permite şi crearea de reguli de securizare a conexiunilor (*connection security rules*) care controlează autentificarea dintre două calculatoare (două servere dintr-o reţea, spre exemplu) pentru a asigura faptul că orice conexiune stabilită între aceste calculatoare este una securizată, folosind diferite metode, precum certificatele.

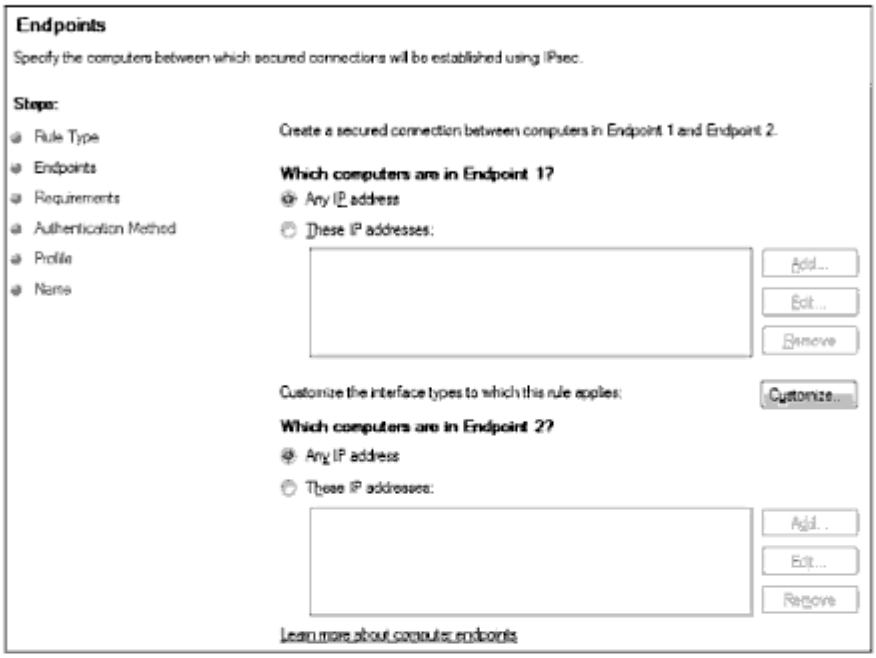
În *Windows Firewall* nu sunt reguli de securizare a conexiunilor configurate în mod implicit. Ele pot fi create explicit de către administrator şi pot fi împărţite în următoarele categorii:

- **Isolation**: Regula restricţionează conectarea la un anumit calculator (deci îl „izolează") pe baza unor criterii de autentificare, precum apartenenţă la un domeniu sau a unor diferite politici de securitate implementate.
- **Authentication exemption**: Regula poate permite accesul fără informaţii de autentificare al altor calculatoare la calculatorul propriu. Acordarea dreptului de conectare se face pe bază de adresă IP.
- **Server to server**: Regula este folosită pentru a realiza o conexiune securizată între două servere. Este nevoie de specificarea serverelor care vor fi implicate în conexiune şi de configurarea autentificării ce se doreşte a se realiza între ele.
- **Tunnel**: Regulă pentru controlul parametrilor unei conexiuni securizate între două puncte, peste o reţea publică, nesecurizată. Se specifică cele două puncte ale conexiunii (*endpoints*) şi metoda de autentificare.
- **Custom**: Regulă complet configurabilă.

Pentru a crea o astfel de regulă se selectează *Connection Security Rules* din panoul stâng şi se face clic pe *link*-ul *New Rule* din panoul de acţiuni, după care se urmează etapele următoare:

Material cu drept de autor

1. Pentru început, se alege tipul regulii ce se doreşte a fi creată. Pentru a avea acces la toate opţiunile, pentru exemplul de faţă se va alege tipul *Custom*.
2. Următoare opţiune cere configurarea capetelor conexiunii (*endpoints*). *Endpoint*-ul 1 poate fi calculatorul local sau o subreţea de adrese IP ce pot fi atribuite calculatorului local, iar *endpoint*-ul 2 va fi cealălalt capăt al conexiunii, specificat printr-o adresă IP sau un interval.



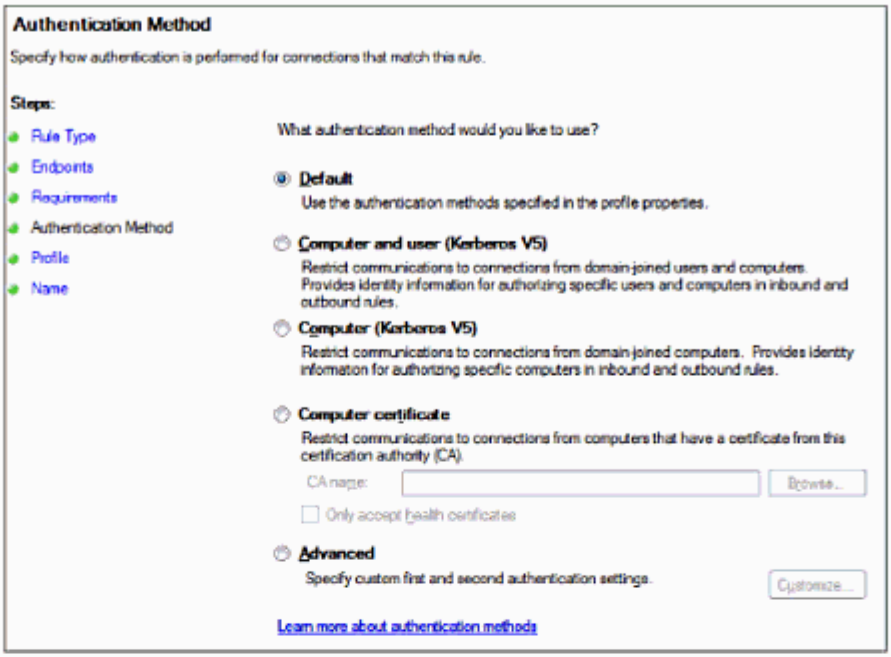
6-6: Definirea capetelor unei conexiuni securizate

3. În pagina următoare se selectează tipul de autentificare ce va fi folosită:
   - **Request authentication for inbound and outbound connections**: Autentificarea nu este obligatorie, dar este de preferat. Conexiunile *inbound* şi *outbound* nesecurizate vor reuşi dar dacă se doreşte utilizarea unei autentificări se poate realiza acest lucru.
   - **Require authentication for inbound connections and request authentication for oubound connections**: Conexiunile iniţiate în exterior trebuie să fie autentificate, dar pentru cele iniţiate local este opţional.
   - **Require authentication for inbound and outbound connections**: Atât conexiunile iniţiate din exterior cât şi cele iniţiate local trebuie autentificate, altfel regula va bloca realizarea lor.
   - **Do not authenticate**: Nu este necesară autentificarea.

4. În următoarea pagină se selectează metoda de autentificare folosită de regulă. Opţiunea *Default* foloseşte autentificarea implicită a profilului. Se mai pot selecta metode de autentificare bazate pe utilizator şi calculator (ceea ce necesită apartenenţă la un domeniu), doar în funcţie de calculator, sau pe baza unui certificat. Prin opţiunea *Advanced* se pot specifica două sesiuni de autentificări, secvenţiale, fiecare prin metodele sale.
5. Pe pagina *Profile* se aleg profilurile pentru care regula va fi activă.
6. În ultima pagină se introduce un nume şi o descriere a profilului.

După creare, regula apare în lista de *Connection Security Rules* şi i se pot modifica proprietăţile ca şi în cazul regulilor *Inbound* sau *Outbound*.

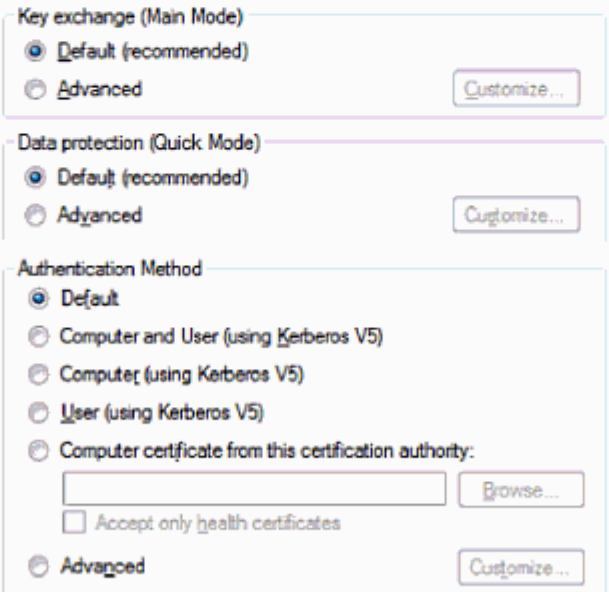
Material cu drept de autor



6-7: Definirea metodelor de autentificare

6.4.1.5 Configurări IPSec

IPSec (*IP Security Protocol*) reprezintă o serie de servicii şi protocoale de securitate orientate spre asigurarea confidenţialităţii datelor transferate în interiorul unei reţele sau prin conexiuni de tip VPN. Avantajul major al *IPSec* este că datele pot fi securizate indiferent dacă dispozitivele de pe parcurs suportă sau nu aceste protocoale. Criptarea în *IPSec* se face separat pentru fiecare pachet iar ca metode de autentificare pot fi folosite certificatele digitale.

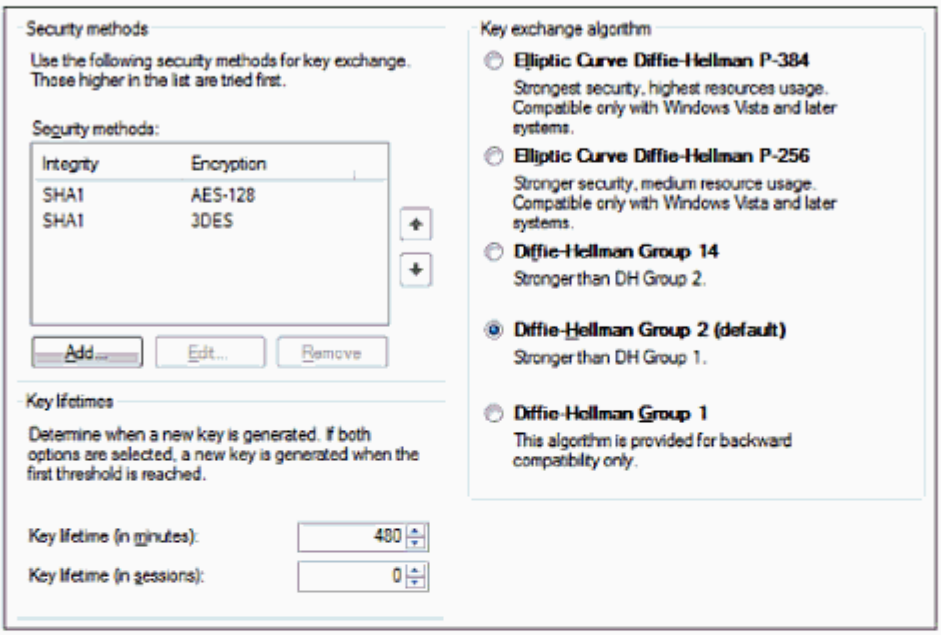


6-8: Setări IPSec

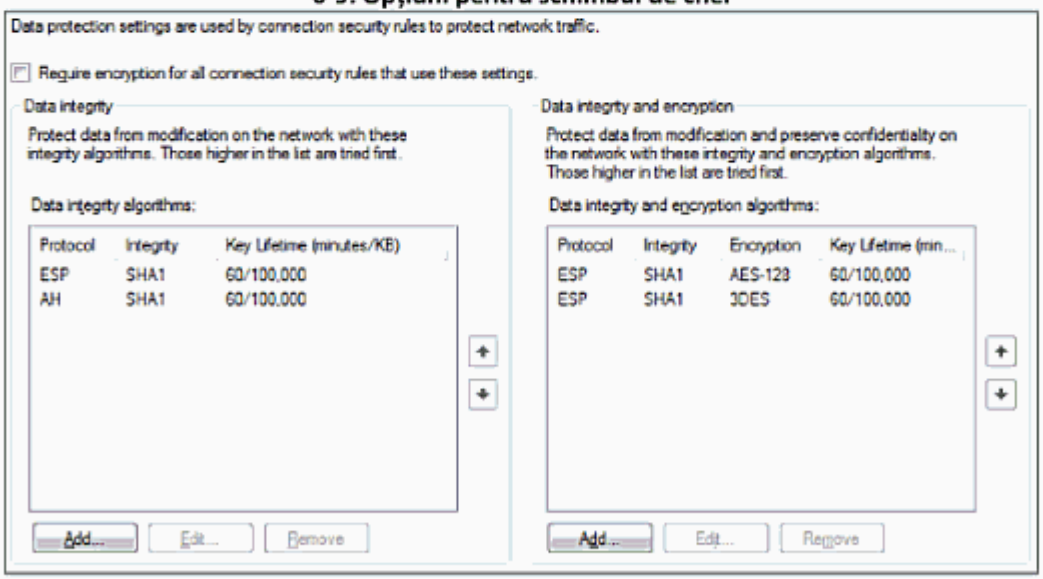
În *Windows Server 2008*, setările ce adresează *IPSec* sunt incluse în *Windows Firewall* şi sunt accesibile la pagina *IPSec Settings* din cadrul ferestrei de proprietăţi a *firewall*-ului. Modificarea setărilor implicite sunt accesibile prin apăsarea butonului *Customize*. Acestea vor afecta toate regulile de tip *Connection Security Rule* definite în *firewall*.

Configurările pentru *IPSec* se încadrează în trei categorii: schimbul de chei (*key exchange*), protejarea datelor (*data protection*) şi modalitatea de autentificare (*authentication method*).

Pentru a schimba modul în care se realizează schimbul de chei, se alege optiunea *Advanced* din grupul *Key Exchange* şi se face clic pe butonul *Customize*.



6-9: Opţiuni pentru schimbul de chei



6-10: Opţiuni pentru protecţia datelor

Modul implicit foloseşte algoritmul *Diffie-Hellman Group 2* (bazat pe o cheie publică şi una privată). Există posibilitatea optării pentru un algoritm mai puternic, ca *Elliptic Curve Diffie-*

Hellman P-384. Trebuie avut în vedere şi faptul că selectarea acestui algoritm impune clientului restricţia ca acesta să ruleze cel puţin Windows *Vista* iar serverul *Windows Server 2008*. Tot aici poate fi modificată şi durata de viaţă a cheilor. Teoretic, securitatea unei chei este invers proporţională cu durata sa de viaţă.

Tot din fereastra principală a setărilor *IPSec* poate fi modificată şi metoda folosită pentru asigurarea securităţii datelor (criptare). Pentru aceasta se selectează opţiunea *Advanced* din categoria *Data protection* (*Quick Mode*) şi se apasă butonul *Customize*.

Implicit se folosesc doi algoritmi pentru a asigura integritatea şi securitatea datelor: ESP şi AH. Protocolul *Encapsulating Security Payload* (ESP) oferă autentificarea sursei datelor, integritate şi protecţie împotriva atacurilor de tip *replay* pentru conţinutul pachetelor IP. *Protocolul Authentication Header* (AH) oferă securitate pentru antetul IP.

Ultima categorie de opţiuni se referă la metoda de autentificare folosită pentru a realiza conexiuni securizate:

- **Computer and User (Using Kerberos V5)** autentifică atât calculatorul cât şi utilizatorul. *Kerberos V5* foloseşte un sistem de chei sau tichete atribuite calculatoarelor din domeniu. Mesajele trimise de aceste calculatoare sunt autentificate prin tichet care este ataşat în date.
- **Computer (Using Kerberos V5)** autentifică doar calculatorul.
- **User (Using Kerberos V5)** autentifică doar utilizatorul.
- **Computer certificate from this certification authority** necesită specificarea unui CA (*Certificate Authority*) iar autentificarea se face baza certificatelor digitale

6.4.2 Monitorizare

Un aspect important al administrării oricărei reţele îl reprezintă monitorizarea serverelor şi a traficului din reţea. În *Windows Server 2008*, monitorizarea performanţei serverului este realizată de către utilitarul *Reliability and Performance Monitor*. Pe de altă parte, un alt utilitar, *Event Viewer*, permite monitorizarea jurnalelor menţinute pentru a ajuta la identificarea problemelor ce pot apărea în funcţionarea pe termen lung a serverului.

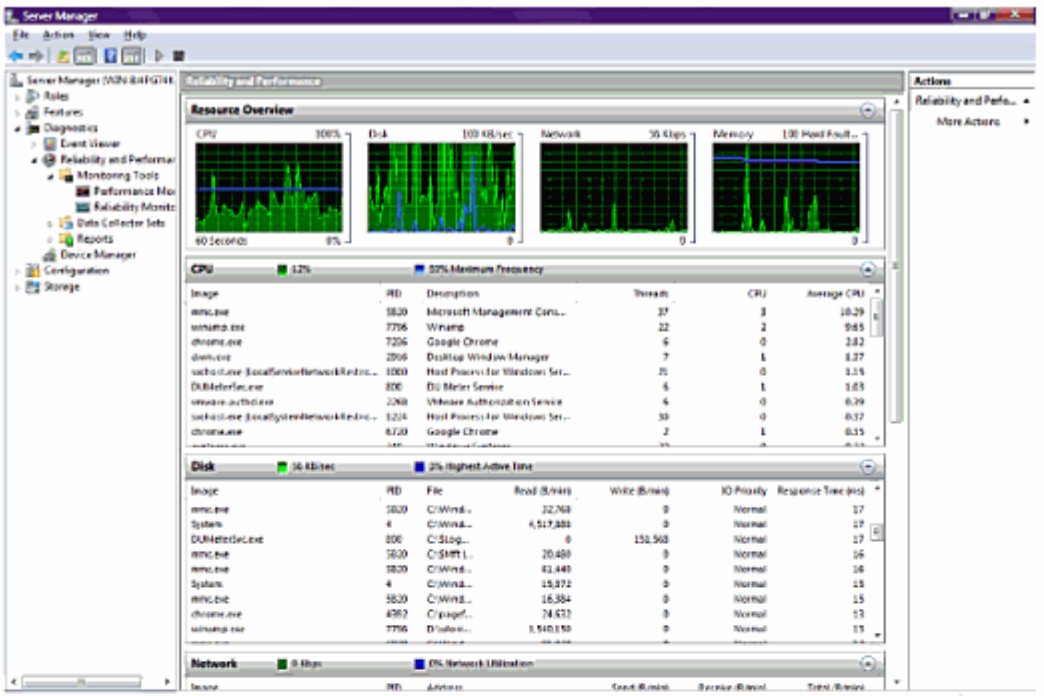
6.4.2.1 Reliability and Performance Monitor

Reliability and Performance Monitor este un utilitar ce permite monitorizarea în timp real a stării serverului atât *hardware* cât şi pe partea de aplicaţii. De asemenea, el poate crea rapoarte de perfomanţă şi alerte pentru valori critice.

În mod intuitiv, performanţa (*performance*) descrie cât de repede serverul execută anumite sarcini în timp ce siguranţa (*reliability*) este o măsură a frecvenţei cu care serverul execută o sarcină exact aşa cum ar trebui, conform configuraţiei sale. *Reliability and Performance Monitor* oferă o multitudine de informaţii cu privire la modul în care atât *hardware*-ul cât şi *software*-ul funcţionează (inclusiv sistemul de operare în sine). În general, monitorizarea performanţei are ca scop identificarea elementelor care încetinesc viteza sistemului şi a motivelor pentru care acestea funcţionează posibil necorespunzător. Pe de altă parte, situaţiile neprevăzute sau necontrolate, cum ar fi dispozitive care nu se iniţializează sau se iniţializează incorect, precum şi servicii oprite sau restartate la momente necorespunzătoare intră în categoria excepţiilor de *reliability*.

Toate utilitarele de diagnostic oferite de *Windows Server 2008* pot fi accesate prin *Server Manager*, din categoria *Diagnostics*.

Material cu drept de autor



6-11: Reliability and Performance Monitor

În partea superioară a *Reliability and Performance Monitor* sunt afişate informaţii sumare cu privire la încărcarea procesorului, discului, a memoriei şi a reţelei. În partea inferioară, fiecare resursă este detaliată după cum urmează:

- **CPU**: Sunt afişate ocuparea procesorului (sau procesoarelor) precum şi frecvenţa maximă. De asemenea, utilizarea sa este detaliată pentru fiecare aplicaţie în parte, dupa PID, nume, număr de fire de execuţie din aplicaţia respectivă, procentajul din procesor folosit şi încărcarea medie de-a lungul timpului.
- **Disk**: Este afişat fluxul total de date la citire/scriere din secunda curentă. Statisticile detaliate pentru fiecare proces cu privire la utilizarea discului, cuprind: numele procesului, PID-ul său, fişierul curent aflat în scriere sau citire, vitezele curente de citire şi scriere prioritatea procesului respectiv pentru acces la disc şi valoarea medie a timpului de răspuns al discului.
- **Network**: Este afişat traficul total prin interfeţele de reţea la momentul curent. Pentru fiecare proces, identificat prin nume şi PID sunt detaliate: adresa destinaţiei cu care aplicaţia schimbă date la momentul respectiv, datele trimise şi primite şi lărgimea de bandă totală utilizată.
- **Memory**: Se afişează global numărul de *page faults*[1] pe secundă şi procentajul de memorie utilizată la momentul respectiv. Pentru fiecare proces în parte, se ţine evidenţa: numărului de *hard faults*-uri pe minut, a *working set*-ului, adică a cantităţii totale de memorie folosita de acea instanţă a aplicaţiei, memoria partajabilă, care poate fi accesată şi de către alte aplicaţii şi memoria privată.

Din cadrul *Reliability and Performance Monitor*, utilitarul *Reliability Monitor* oferă o perspectivă globală asupra evenimentelor din sistem ce au afectat de-a lungul timpului funcţionarea sa într-un mod negativ.

[1] Un page fault reprezintă o situaţie în care se accesează date din memoria virtuală a unui proces dar care nu se găsesc în memoria fizică şi trebuie aduse din fişierul de paginare.

Material cu drept de autor

6.4.2.2 Performance Monitor

Performance Monitor reprezintă un utilitar grafic pentru măsurarea performanţei sistemului şi a altor calculatoare din reţea. *Performance Monitor* scanează performanţa echipamentelor fizice, ca procesorul, discul şi memoria, fiecare element ce poate fi analizat fiind considerat un obiect.

Un anumit parametru ce este măsurat pentru un anumit obiect este denumit un *counter*. Spre exemplu, pentru procesor, pot fi măsurate *counter*-e ca procentajul utilizat sau numărul de întreruperi pe secundă. *Performance Monitor* poate afişa informaţiile sub formă de grafice, în diferite configuraţii.

Adăugarea unor noi *counter*-e se poate face fie din meniul contextual al graficului, fie prin butonul *Add* de deasupra graficului. La adăugarea unui nou *counter*, se poate selecta maşina proprie sau o alta din reţea, resursa care va fi urmărită şi parametrul specific (*counter*-ul) conform căruia se va realiza graficul.

6.4.2.3 Event Viewer

Fiecare instanţă a unei acţiuni ce este executată într-un sistem este considerată un eveniment. *Event Viewer*, spre deosebire de alte utilitare de monitorizare, nu oferă informaţiile în timp real, ci permite accesarea şi interpretarea jurnalelor în care sunt trecute de-a lungul timpului detalii despre momentul şi modul în care aceste evenimente au avut loc.

Event Viewer poate fi accesat tot din *Server Manager*, de la categoria *Diagnostics*. *Event Viewer* încadrează jurnalele în două categorii: **Windows Logs** şi **Application and Services Logs**. Jurnalele de tip *Windows Logs* includ următoarele:

- **Application log**: jurnal ce înregistrează evenimentele diverselor aplicaţii ce rulează în sistem. De regulă aceste evenimente sunt controlate din codul aplicaţiei. Tot aici sunt încadrate şi alertele definite în *System Monitor*.
- **Security log**: jurnal ce monitorizează evenimentele legate de drepturile de accesare a fişierelor, de autentificarea utilizatorilor, de apartenenţa la un domeniu, etc.
- **Setup log**: jurnal constituit din evenimentele de la instalarea şi configurarea aplicaţiilor. De asemenea, evenimentele legate de adăugarea sau eliminarea unor roluri ale serverului, precum şi eventualele erori sau avertismente din timpul acestora sunt înscrise aici.
- **System log**: jurnal ce ţine evidenţa anumitor evenimente predefinite în Windows, cum sunt cele legate de instalarea sau funcţionarea incorectă a driverelor şi tot ceea ce ţine de servicii şi performanţa sistemului.

Cealaltă categorie de jurnale, *Application and Services Logs*, păstrează informaţii particulare pentru aplicaţii şi componente ale serverului. Aceste jurnale includ evenimente de tipul *Hardware Events* (instalări, erori), *Internet Explorer Events* şi *Key Management Services* (evenimente legate de folosirea cheilor pentru criptarea informaţiile trimise sau primite din reţea).

Pentru întreţinere, jurnalele pot fi golite periodic sau salvate în fişiere pe disc în diferite formate.

Material cu drept de autor

Întrebări

1. Care dintre rotocoalele de mai jos permite un transfer sigur de fişiere?

- *SSH-1*
- *TFTP*
- *SCP*
- *Telnet*

2. Care din echipamentele de mai jos pot fi folosite pentru a contracara atacuri?

- *firewall-urile*
- *IDS-urile*
- *concentratoarele VPN*
- *atât IDS-urile cât şi firewall-urile*

3. Care din protocoalele de mai jos nu are nevoie de inspectare a traficului pentru a funcţiona printr-un firewall?

- *FTP*
- *IRC*
- *WWW*
- *SQL*

4. De ce nu are sens următoarea regulă?

```
iptables -A INPUT --mac-source 01:01:01:01:01:01 -j REJECT
```

- *selectarea după adrese MAC nu se poate face pe lanţul INPUT*
- *regula are sens*
- *adresa MAC este incorectă*
- *nu se poate folosi REJECT cu adrese MAC*

5. Ce efect are următoarea regulă?

```
iptables -A INPUT -p ICMP --icmp-type echo-request -s 192.168.1.0/24 -m limit 3/s -j ACCEPT
```

- *regula este incorectă sintactic*
- *se primesc pachete ICMP de tip echo-request de la 192.168.1.0/24, dar la o rată de 3 pe secundă*
- *se primesc pachete ICMP de tip echo-request de la 192.168.1.0/24 dar se scriu în log doar 3 pe secundă*
- *nu se primesc pachete ICMP de tip echo-request de la 192.168.1.0/24 şi se scriu în log doar 3 pe secundă*

6. Care din următoarele utilitare pot fi folosite ca şi IDS-uri

- *Snort*
- *Wireshark*
- *Iptables*
- *nat*

Material cu drept de autor

7 DNS

"You know it's love when you memorize her IP address to skip DNS overhead"

Ce se învață din acest capitol?

- Ce sunt domeniile de nume
- Ce sunt serverele DNS
- Configurarea serviciului DNS pentru clienti
- Configurarea BIND
- Configurarea rolului de server DNS pe Windows

Cine este...

Paul Mockapetris *este inventatorul DNS. Mockapetris a lucrat ca manager de program la diverse companii de rețelistică și Internet. Din 2003, activează la Institutul de Științe Informatice (ISI) de la Universitatea din California de Sud.*

Paul Vixie *este autorul unui număr important de RFC-uri și utilitare standard UNIX. A fost creatorul serverului DNS bind și arhitectul acestuia până la versiunea 8. Operează serverul rădăcină L din DNS.*

7.1 Protocolul DNS

Pentru a facilita accesarea resurselor in Internet este necesară existența unei asocieri între adresa IP a acestora și un nume ușor de reținut. Asta deoarece este imposibil ca cineva să poată reține toate adresele IP folosite pe glob. La începutul Internetului stațiile erau accesate pe baza intrărilor din fișierul `/etc/hosts` care făceau **translatarea** adreselor din formatul preferat de utilizatorul uman (ex: `www.wikipedia.org`) în adrese IP necesare echipamentelor de rețea.

O dată cu creșterea explozivă a dimensiunii Internetului, a devenit evident că era nevoie de o soluție care să rezolve problemele de scalabilitate. Soluția pentru aceste probleme a fost dezvoltarea unui nou protocol, *Domain Name Server* (DNS). Protocolul a fost dezvoltat în anii '80, propus ca RFC și adoptat apoi ca standard Internet. Deși protocolul are peste 20 de ani vechime, există și îmbunătățiri și facilități noi aduse serviciului de nume (extensii de securitate, stocarea certificatelor digitale în DNS, etc.).

DNS este în esență o **bază de date distribuită** care asociază diferite informatii cu domeniile DNS. Noutatea pe care o aducea DNS la vremea propunerii sale ca standard nu era atât conceptul de bază de date distribuită - în sensul în care informațiile din baza de date sunt păstrate pe stații diferite - ci mai degrabă faptul că și **administrarea bazei de date urma să se facă distribuit**.

7.1.1 Domenii DNS

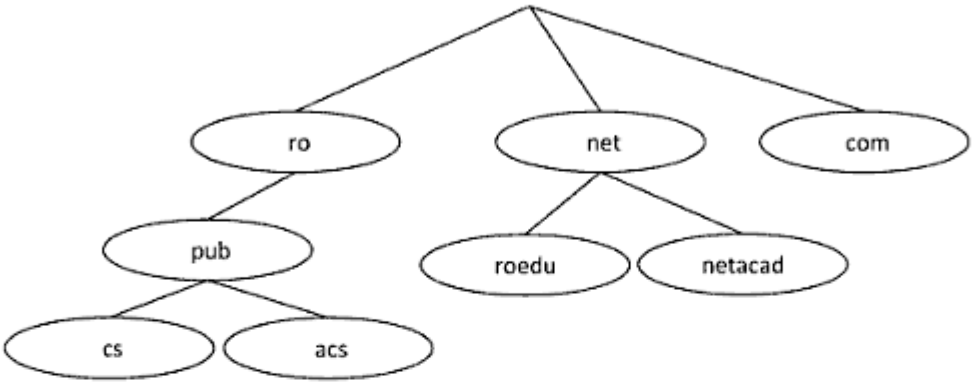
*Un **domeniu DNS** este o grupare a mai multe stații și servere ce au un sistem de administrare comun și sunt identificate de un nume unic.*

Între domenii există o legătură ierarhică. În general, un domeniu are în componență alte subdomenii și face parte dintr-un alt domeniu la rândul lui (care ar putea fi numit supradomeniu, deși termenul nu este folosit în literatura de specialitate). Această relație ierarhică de incluziune poate fi cel mai simplu explicată print-un exemplu concret: subdomenii pentru domeniul `pub.ro` sunt `cs.pub.ro`, `electronica.pub.ro` sau `acs.pub.ro`, iar supradomeniul pentru `pub.ro` este `ro`. Deși între domenii există această relație de incluziune,

Material cu drept de autor

trebuie reținut faptul că subdomeniile nu trebuie să aibă fiecare același administrator ca și domeniul din care fac parte. După cum s-a mai spus, noutatea DNS constă în posibilitatea administrării bazei de date în mod distribuit și din această cauză, în general, domeniile au delegate pentru un subdomeniu un nou administrator.

Datorită structurii sale ierarhice, baza de date DNS poate fi vizualizată ca un arbore multicăi, în care nodurile sunt domenii. De exemplu, pentru domeniile enumerate în paragraful precedent, arborele asociat este prezentat în figură.



7-1: Ierarhia DNS

După cum se observă în figură, fiecare nod, mai puțin nodul rădăcină, are asociat un nume precum co, acs, pub, ro. Această modalitate de numire, în care nu se specifică numele complet al domeniului este numită **referire relativă** (*relative domain name*). Dacă se deține doar numele relativ al unui domeniu, **numele complet** al domeniului (*fully qualified domain name - FQDN*) poate fi aflat prin concatenarea numelor supradomeniilor aflate în drumul de la domeniu la rădăcină și folosirea punctului ca separator între numele domeniilor.

7.1.1.1 Domenii speciale

Principalul rol al serviciului de nume este de a asocia nume stațiilor din Internet, care altfel ar fi fost identificate de adrese IP. Aceasta înseamnă că serviciul DNS va trebui să ofere utilizatorilor cel puțin două operații: **rezolvare directă** (*resolve lookup*) - aflarea adresei IP atunci când știm numele stației și **rezolvare inversă** (*reverse lookup*) - aflarea numelui unei stații atunci când știm adresa IP.

Ierarhia DNS începe cu câteva domenii speciale, denumite TLD (*top level domains*).

com	-	domenii folosite de organizațiile comerciale
edu	-	domenii folosite de organizațiile educaționale
gov	-	domenii folosite organizațiile guvernamentale
mil	-	domenii folosite organizațiile militare
org	-	domenii folosite organizațiile non-profit
net	-	domenii ale organizațiilor ce administrează rețele mari
ro, fr, eu, us	-	domenii de țară

Aceste domenii sunt administrate de către ICANN (*Internet Corporation For Assigned Names and Numbers*). Ele sunt subdomenii ale unui domeniu generic, fără nume, care este rădăcina ierarhiei. Toate domeniile din Internet sunt până la urmă subdomenii ale acestor domenii din vârful ierarhiei. Înregistrarea unui subdomeniu .eu costă aproximativ 14 euro pe

Material cu drept de autor

an, iar un subdomeniu `.ro` costă ceva mai mult de 50 USD + TVA (19%), dar este înregistrat pe viață. De la 1 ianuarie 2007, persoanele fizice și juridice din Romania pot trimite cereri pentru înregistrarea de domenii `.eu`. Potrivit EURid (European Registry for Internet Domains), până la data de 20 septembrie 2007, România „a contribuit" cu 11,851 de nume de domenii `.eu` active. Comparativ, locuitorii Germaniei dețin 822,712 de domenii `.eu`, iar cei ai Republicii Cehe dețin 53,830 de domenii.

După cum s-a precizat mai sus, protocolul DNS oferă și posibilitatea de *reverse-lookup*. Această facilitate este folosită de utilitare de *troubleshooting* precum `traceroute` sau `ping`. Pentru a păstra o consistență în funcționare, și traducerea inversă se face tot cu ajutorul unui domeniu, care poartă numele de **in-addr.arpa**. El nu este un *top-level domain* și a fost creat pentru a permite rezolvare inversă, din adrese IP în nume.

Acesta conține subdomenii care corespund cu octeții dintr-o adresă IP, în ordine inversă. De exemplu, pentru adresa 141.85.37.1, cererea pentru traducerea inversă se va face pentru `1.37.85.141.in-addr.arpa`. În baza de date DNS acestei intrări îi corespunde numele stației.

7.1.1.2 Cereri DNS

S-a stabilit deci nevoia pentru protocolul DNS și structura ierarhiei acestuia. Ce se întâmplă însă în momentul efectuării unei cereri DNS? Cel mai adesea o cerere va fi efectuată de către un *browser* web la introducerea unei adrese URL, ca `www.google.com`. Clientul web va trebui să afișeze conținutul paginii care se află pe serverul HTTP referit de acest nume. Pentru a putea face acest lucru, trebuie trimis un mesaj HTTP de tip *GET*. Ca orice pachet ce urmează să fie trimis în Internet, și un mesaj HTTP va trebui să conțină în antentul de nivel 3, o adresă IP destinație. Aici intervine clientul de DNS (integrat în *browser*) care face o cerere DNS pentru a putea afla adresa IP a serverul HTTP care servește paginile pentru domeniul `www.google.com`. Cererea va fi făcută către serverul specificat pe stație, ca fiind serverul local de DNS. Pe un sistem Linux acest server este specificat în fișierul `/etc/resolv.conf`. Bineînțeles că serverul DNS local s-ar putea să nu cunoască adresa IP pentru `www.google.com`, căci după cum s-a specificat anterior, baza de date DNS este distribuită și administrată distribuit.

Cererile trimise de clienții DNS, serverelor locale poartă o denumire specială în terminologia DNS, și anume **cereri recursive**. Această denumire este dată de faptul că serverul DNS local este obligat să rezolve cererea indiferent dacă are informații despre ea sau nu, prin interograrea altor servere DNS.

Pentru a putea simplifica protocolul, doar clienții de DNS pot face cereri recursive. Dacă serverul local nu știe sa răspundă cu o adresă IP pentru un anumit domeniu, el va face o cerere **nerecursivă** la un alt server DNS *remote* (această operație se va detalia în următorul subcapitol). Cererea este numită nerecursivă din cauza faptului că dacă serverul *remote* interogat nu poate rezolva numele de domeniu, acesta va oferi un răspuns negativ (fără a mai încerca el să întrebe alt server de nume) alături de un *hint* care să indice un alt server DNS care ar putea să traducă numele. În continuare, serverul DNS local, deși a primit un răspuns negativ, deoarece a primit anterior o cerere **recursivă**, va continua să întrebe alte servere DNS *remote*, până va primi un răspuns pozitiv pe care îl va returna clientului.

Pentru a concluziona, diferența dintre o cerere recursivă și una nerecursivă, este că prima dintre acestea va fi întotdeauna rezolvată, pe când a doua, nu (poate primi răspuns negativ).

7.1.2 Tipuri de servere DNS

S-a putut observa că fiecare domeniu DNS are o entitate administrativă. Această entitate administrativă este un **server DNS** sau, conform terminologiei DNS, un **server de nume**.

Material cu drept de autor

Principalul rol al serverului de nume asociat unui domeniu este de a răspunde la cereri despre stațiile şi serverele aflate în gestiunea sau autoritatea sa.

*Se spune că un server de nume este **server autoritar** pentru o intrare din baza de date DNS dacă intrarea face parte din domeniul gestionat de serverul de nume.*

***Atenție!** Faptul că un server este autoritar pe domeniul `cs.pub.ro`, nu înseamnă că el nu va încerca să rezolve o cerere recursivă pentru adresa `ubuntuforums.org`, ci doar faptul că adresa `cs.pub.ro` va putea fi tradusă întotdeauna direct de către el.*

După cum se va vedea în continuare, rezolvarea unui domeniu, a unui nume sau a unei adrese este un proces iterativ ce poate necesita interogarea mai multor servere DNS, şi are, deci, o latență considerabilă. Din această cauză, în general, serverele de DNS vor răspunde şi la cereri care nu intră în autoritatea lor (nu fac parte din domeniul gestionat), din două motive: acest lucru va simplifica implementarea clientului de DNS şi în acelaşi timp se va putea folosi un **cache** pentru toate stațiile ce folosesc acel server de nume. Astfel, clientul va trimite cererea DNS serverului indiferent dacă cererea se referă la un nume din domeniul serverului sau nu; acesta va rezolva cererea fie local, dacă numele face parte din domeniul său, fie prin interogarea iterativă a mai multor servere de nume, după cum urmează. Odată aflat răspunsul acesta va putea fi păstrat în *cache*, şi cererile ulterioare vor fi servite din *cache*.

Datorită *cache*-ului se poate întâmpla ca modificările operate de serverul de nume asupra porțiunii sale din baza de date DNS să nu fie vizibile imediat. Aceasta datorită faptului că alte servere de nume vor folosi de obicei *cache*-ul pentru a întoarce răspunsuri clienților, răspunsuri care pot fi neactualizate. Pentru a minimiza latența propagării schimbărilor efectuate, administratorul domeniului poate specifica durata maximă de timp pentru care un răspuns trimis poate sta în *cache*-ul altui server de nume. Trebuie reținut, însă, faptul că o valoare de ordinul a câtorva ore nu este exagerată, şi reprezintă o practică destul de comună în Internet. Pe de altă parte, modificările în baza de date DNS nu sunt atât de dese, astfel că această politică are sens, mai ales datorită faptului că folosirea *cache*-ului DNS diminuează foarte mult latența rezolvării numelor.

Convergența DNS este unul dintre cele mai lente procese din Internet.

7.1.2.1 Server DNS Caching-only

Problema cu cererile recursive este că, în cantitate mare, pot îngreuna simțitor un server DNS. De aceea, este considerată o practică bună ca un server DNS să accepte cereri doar din partea rețelelor locale aflate în domeniul pentru care acest server este autoritar. Dacă administratorul ar permite cereri recursive din tot Internetul, severul său ar putea fi foarte uşor atacat.

În concluzie, chiar dacă o rețea nu are un domeniu DNS (pentru că sunt folosite adrese private de exemplu), un server DNS este totuşi necesar, deoarece clienții DNS nu trimit decât cereri recursive, care trebuie rezolvate undeva. În plus, un server DNS este util şi pentru *cache*-ul pe care îl menține. Din această cauză există şi servere de nume *caching-only*. Ele sunt servere de nume, care rezolva cereri recursive, dar care nu sunt servere autoritare pentru niciun domeniu.

***Notă**: dacă un server nu este autoritar peste niciun domeniu (caching-only), înseamnă că răspunsul la o cerere recursivă va fi dat întotdeauna din cache, sau din rezultatul pozitiv al unei cereri nerecursive efectuate de serverul caching-only.*

Material cu drept de autor

7.1.2.2 Servere DNS rădăcină

Serverele rădăcină *sunt servere de nume administrate de InterNIC, şi care gestionează o parte din domeniile top-level.*

Acestea sunt cunoscute de serverele de nume în mod implicit şi sunt de obicei interogate în mod nerecursiv de către un server local, atunci când acesta nu este autoritar pentru domeniul cerut şi nici nu posedă informaţia în *cache*. Dacă serverul rădăcină nu ştie să traducă cererea, va trimite serverului local un răspuns negativ, alături de un *hint* care îi va sugera un alt server DNS care ar putea şti să translateze numele interogat.

7.1.2.3 Servere DNS forwarder

Dacă un server de nume trebuie să răspundă la o cerere recursivă de la un client, acesta va încerca mai întâi să vadă dacă numele interogat face parte din domeniul pentru care el este autoritar. Dacă acesta nu e autoritar peste domeniul cerut, va încerca să caute intrarea respectivă în *cache*. În cazul în care nu o găseşte în *cache*, serverul va trebui să facă cereri nerecursive către alte servere, pentru a putea realiza rezolvarea. În mod normal, serverul va interoga un server rădăcină. Aici intervine noţiunea de server **forwarder**.

Un administrator poate configura pe serverul său DNS, adresa IP a unui server special care va avea un rol de *forwarder* pentru acesta. Mai exact, în situaţia de mai sus, în loc ca serverul local să apeleze la un server rădăcină, va apela mai întâi la serverul *forwarder* pe care îl are configurat. *Forwarder*-ul va consta de obicei dintr-un server *chaching-only* care va fi cu atât mai bun, cu cât e folosit mai des şi de mai multe servere. Dacă *forwarder*-ul nu va reuşi să dea un răspuns pozitiv la cererea nerecursivă a serverului local, se va apela la un server rădăcină cunoscut.

Observaţie: Ca să fie eficient (să aibă informaţie în cache), forwarder-ul va trebui să primească şi cereri recursive.

7.1.2.4 Servere Master/Slave

Din motive de redundanţă şi de distribuire a încărcării, pot exista mai multe servere autoritare pentru acelaşi domeniu. Cu toate acestea, doar unul dintre ele va fi server *master*, celelalte vor fi servere *slave*.

Serverul master *poate şterge, adăuga sau modifica intrări din porţiunea sa din baza de date DNS.* ***Serverele slave*** *vor transfera la pornire informaţiile de la serverul* master.

În continuare, periodic, serverele *slave* vor interoga **seria bazei de date** de la serverul *master*. Dacă seria de la serverul *master* este mai mare decât seria curentă a serverului *slave*, acesta va transfera din nou baza de date de la serverul *master*. Pentru a reduce latenţa propagării schimbărilor la serverele *slave*, protocolul DNS prevede, de asemenea, mecanisme de notificare a serverelor *slave*. Atunci când este necesar, în general la pornirea sau repornirea serverului *master*, acesta poate notifica serverele *slave* trimiţându-le seria bazei de date.

7.1.3 Tratarea unei cereri DNS

Protocolul DNS specifică existenţa a două tipuri de răspunsuri la cererile DNS: **răspunsuri autoritare** şi **răspunsuri neautoritare**. Doar serverele ce gestionează un domeniu (serverele *master* şi *slave* ale domeniului) pot trimite răspunsuri autoritare, şi asta doar pentru staţiile din domeniul gestionat.

Material cu drept de autor

7.1.3.1 Tratarea unei cereri recursive

În continuare se vor sumariza paşii principali ai procesului de tratare a unei cereri recursive:

- Clientul face o cerere recursivă către serverul DNS local.
- Dacă serverul este autoritar peste un domeniu, analizează numele interogat pentru a îşi da seama dacă este chiar numele domeniului peste care este autoritar. În caz afirmativ, oferă un răspuns autoritar.
- Dacă serverul nu este autoritar peste niciun domeniu sau dacă nu s-a putut da un răspuns autoritar direct, se va căuta în *cache*-ul serverului. Bineînteles că dacă răspunsul este găsit în *cache*, acesta este oferit clientului.
- Dacă informaţia nu a fost găsită în *cache* se va face cel puţin o cerere nerecursivă.

7.1.3.2 Tratarea unei cereri nerecursive

În cazul în care un server local nu ştie să traducă o cerere recursivă, aceasta poate fi rezolvată cu ajutorul unor cereri nerecursive adresate pe rând mai multor servere de nume. O cerere nerecursivă va întoarce un răspuns pozitiv doar dacă serverul interogat are intrarea în *cache* sau este autoritar pentru cerere. Altfel, serverul de nume interogat va răspunde cu un mesaj specificând faptul că răspunsul este necunoscut şi indicând un alt server de nume. În această situţie, serverul interogat decide dacă cererea reprezintă un domeniu inclus în domeniul său de autoritate sau nu, pentru a determina serverul recomandat. Se disting deci 2 cazuri:

1. Dacă cererea este un domeniu inclus în domeniul serverului, se caută în *cache* şi va fi indicat serverul pentru cel mai specific domeniu al cererii. Dacă niciunul dintre domeniile specifice nu se află în *cache* se va da un răspuns autoritar ce poate fi pozitiv sau negativ. Spre exemplu, dacă cererea `eglab.rc.cs.pub.ro` ajunge la serverul autoritar pentru domeniul `pub.ro`, mai întâi va fi căutat în *cache* numele complet `eglab.rc.cs.pub.ro`. Dacă acesta nu este găsit, se va căuta `rc.cs.pub.ro`. Dacă şi această căutare a eşuat se va căuta adresa `cs.pub.ro` în fişierele de configuraţie locale. Dacă aceasta există, răspunsul va fi pozitiv, altfel răspunsul va fi răspuns autoritar negativ.
2. Dacă cererea nu este inclusă în domeniul serverului interogat, acesta va indica serverul din *cache* pentru cel mai specific domeniu al cererii. Astfel, pentru cererea `orange.csl.cmu.edu` va fi căutat în *cache* mai întâi numele complet. Dacă acesta nu este găsit se va căuta `csl.cmu.edu`, apoi `cmu.edu` şi în final doar domeniul `edu`. Dacă niciuna dintre aceste căutări nu s-a încheiat cu succes va fi indicat un server rădăcină.

7.1.3.3 Exemplu de rezolvare a unei cereri

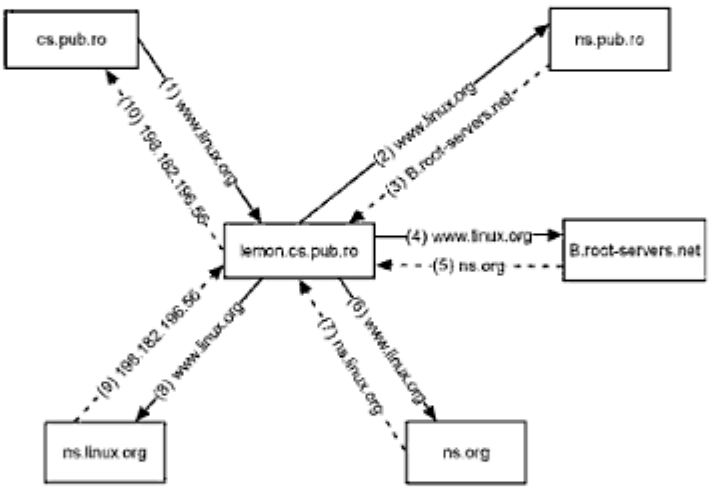
Pentru exemplificare, se consideră o aplicaţie ce trebuie să rezolve numele `www.linux.org` şi că aplicaţia rulează pe staţia `lemon.cs.pub.ro`, care are configurat ca server de nume serverul `cs.pub.ro`. Paşii urmaţi sunt:

1. Staţia `lemon.cs.pub.ro` trimite o *cerere recursivă* serverului `cs.pub.ro` în care solicită aflarea adresei asociate numelui `www.linux.org`;
2. Serverul va începe prin a analiza apartenenţa numelui `www.linux.org` la domeniul pe care îl gestionează; întrucât nu face parte, se trece la următorul pas;
3. Serverul verifică existenţa adresei în *cache*; se presupune că adresa nu se găseşte în *cache*; în acest caz se trece la pasul următor;
4. Dacă serverul are configurat un server de *forwarding*, atunci va trimite o *cerere nerecursivă* serverului de *forwarding*; în caz contrar va trimite o *cerere nerecursivă* unuia dintre serverele rădăcină; în cazul de faţă, se va considera că serverul `ns.pub.ro` - 141.85.37.8 este configurat ca server de *forwarding*;

Material cu drept de autor

5. Cererea ajunge la serverul `ns.pub.ro` care va căuta în *cache* adresa statiei `www.linux.org`; se presupune că adresa nu se găseşte în *cache*; drept consecinţă, serverul întoarce un *răspuns negativ*, specificând ca *hint* serverul rădăcină `B.root-servers.net - 192.228.79.201`;
6. Serverul cs.pub.ro va trimite cererea serverului rădăcină `B.root-servers.net`; acesta va căuta adresa statiei `www.linux.org` în *cache*; se prespune, din nou, că nu găseşte adresa în *cache*, astfel că va trimite un *răspuns negativ* iar ca *hint* serverul de nume asociat domeniului `.org`, să spunem `ns.org - 216.66.41.146`;
7. Serverul `cs.pub.ro` trimite atunci cererea serverului `ns.org`; acesta va căuta adresa `www.linux.org` în *cache*; se presupune că nu o va găsi; va trimite, deci, un *răspuns negativ*, iar ca *hint* serverul asociat domeniului `linux.org`, să spunem `ns.linux.org`;
8. În continuare, serverul `cs.pub.ro` trimite cererea serverului `ns.linux.org`; acesta, fiind serverul autoritar pentru zona `linux.org`, va căuta adresa `www.linux.org` în baza de date şi va întoarce un *răspuns pozitiv şi autoritar* cu adresa IP asociată;
9. Serverul `cs.pub.ro` întoarce răspunsul staţiei `lemon.cs.pub.ro` şi îl introduce în *cache*.

Pentru a înţelege mai bine, toate cererile şi răspunsurile implicate în rezolvarea cererii au fost reprezentate în figura de mai jos. Cererile sunt reprezentate cu linie continuă, în timp ce răspunsurile sunt reprezentate cu linie punctată. Pentru fiecare cerere sau răspuns a fost reprezentată informaţia solicitată, respectiv oferită şi amprenta de timp corespunzătoare.



7-2: Exemplu de cerere DNS

7.1.4 Structura bazei de date DNS.

În secţiunile precedente, structura bazei de date DNS a fost prezentată simplificat, pentru a explica mai uşor conceptele. S-a văzut că baza de date DNS este de fapt un arbore multicăi în care nodurile sunt reprezentate de domenii, iar frunzele de staţii. În realitate, însă, lucrurile sunt ceva mai complexe. Baza de date DNS este structurată ca un arbore multicăi; totuşi ea nu reţine domenii, staţii şi servere, ci **înregistrări DNS** de forma (cheie, informaţie).

Cheia este reprezentată de un nume complet şi este distribuită în nodurile arborelui. O cheie poate avea asociate mai multe informaţii, în general de tipuri diferite, dar nu obligatoriu. Baza de date DNS este astfel structurată încât cu ajutorul cheii să se localizeze informaţiile asociate.

Înregistrările sunt de mai multe tipuri, grupate după funcţionalitate: înregistrare pentru operaţiile de căutare, înregistrare pentru operaţiile de căutare inversă (*reverse lookup*),

Material cu drept de autor

înregistrare pentru serverele de nume, etc. Fiecare dintre tipurile de înregistrări sunt cunoscute în terminologia DNS după mnemonici, cele mai folosite fiind înregistrările: `A`, `CNAME`, `MX`, `NS`, `PTR`, `SOA`.

7.1.4.1 Înregistrări DNS

În continuare se prezintă descrierea câtorva tipuri de înregistrări din baza de date DNS și rezultatele întoarse la interogarea bazei de date pentru fiecare dintre acestea. Pentru interogare s-a folosit utilitarul `host` prezent în sistemele UNIX.

- `A` identifică înregistrări de tip **adresă**, fiind folosite pentru rezolvarea directă a numelui. Aceste înregistrări asociază chei de tip *nume de stație* de genul `www.kde.org` cu o *adresă IPv4*. Pentru a asocia chei de tip nume de domeniu unor adrese IPv6 se folosesc înregistrări de tip `AAAA`.

```
user@orange:~$ host -t A cs.pub.ro
cs.pub.ro has address 141.85.37.5
```

- `PTR` identifică înregistrări de tip **pointer** și sunt folosite pentru rezolvarea inversă. Acestea asociază chei de tip *adresa IP* de genul 1.37.85.141.in-addr.arpa cu un *nume de domeniu complet*.

```
user@orange:~$ host -t PTR 1.37.85.141.in-addr.arpa
1.37.85.141.in-addr.arpa domain name pointer csr.cs.pub.ro.
```

- `NS` identifică înregistrări de tip **server de nume** (*name server*) și sunt folosite pentru a indica *numele serverelor de nume* autoritate (atât cele *master* cât și cele *slave*, dacă este cazul) asociate *domeniului* specificat în cheie. Înregistrările de tip `NS` sunt folosite pentru **delegarea de subdomenii** către alte servere de nume.

```
user@orange:~$ host -t NS cs.pub.ro
cs.pub.ro name server pub.pub.ro.
cs.pub.ro name server ns.cs.pub.ro.
```

- `MX` identifică înregistrări de tip **server de mail** și sunt folosite pentru a indica *numele serverelor de mail* responsabile pentru mailurile destinate *domeniului* specificat în cheie. Adresa serverelor de mail se specifică cu ajutorul unor înregistrări de tip adresă.

```
user@orange:~$ host -t MX cs.pub.ro
cs.pub.ro mail is handled by 5 mail.cs.pub.ro.
```

- `SOA` identifică înregistrări de tip **start of authority** ce specifică diverși *parametri* pentru **domeniul** indicat în cheie: seria bazei de date, intervalul de timp la care serverul *slave* verifică seria, adresa de mail a administratorului de domeniu, etc.

```
user@orange:~$ host -t SOA cs.pub.ro
cs.pub.ro has SOA record ns.cs.pub.ro. admin.cs.pub.ro. 2007072101 28800 7200 604800
86400
```

- `TXT` identifică o înregistrare de tip **descriere**. Aceasta asociază *numele de stație* indicat de cheie cu un *text de descriere* ASCII.

```
user@orange:~$ host -t TXT cs.pub.ro
cs.pub.ro descriptive text "UPB, Computer Science Departament"
```

- `CNAME` identifică o înregistrare de tip **alias**. Un alias este un *nume de domeniu alternativ* pentru *numele* specificat în cheie. Aliasul va fi asociat cu aceeași adresă cu care este asociată și cheia. De exemplu, considerând aliasul `mail.cs.pub.ro` la numele `prof.cs.pub.ro`, și presupunând că adresa `prof.cs.pub.ro` este `141.85.37.3`, atunci adresa `mail.cs.pub.ro` va fi `141.85.37.3`.

```
user@orange:~$ host -t CNAME mail.google.com
mail.google.com is an alias for googlemail.l.google.com.
```

Material cu drept de autor

Ca o observație cheile folosite în DNS (numele domeniilor, stațiilor, etc.) sunt limitate la caractere alfanumerice (a-z, A-Z, 0-9) și caracterul '-'. Numele de domeniu complete nu pot depăși 255 de caractere, iar numele de domeniu relative nu pot depăși 63 de caractere.

7.2 Configurări de bază DNS

7.2.1 Configurarea clientului DNS pe Linux

7.2.1.1 Fișierul /etc/resolv.conf

În UNIX, informațiile legate de serverele de nume folosite în interogările DNS și alte opțiuni DNS sunt păstrate în fișierul `/etc/resolv.conf`.

Pentru a configura serverul DNS responsabil cu rezolvarea cererilor se adaugă o directivă de tipul `nameserver adresa_ip`. Se pot folosi mai multe servere DNS, pentru fiecare trebuind adaugată o directivă separată:

```
$ cat /etc/resolv.conf
nameserver 88.77.66.55
nameserver 99.88.77.66
```

În cazul în care sunt configurate mai multe servere DNS, se va interoga întotdeauna primul. Celelalte servere se interoghează doar dacă serverul interogat anterior nu răspunde.

7.2.1.2 Fișierul /etc/nsswitch.conf

Pe sistemele UNIX există mai multe metode de rezolvare a numelor: DNS, NIS, `/etc/hosts`, LDAP, etc. Selecția priorității pentru aceste metode de rezolvare a numelor stațiilor (dar nu numai, deoarece la fel pot fi rezolvate și numele utilizatorilor în UID-uri și GID-uri, de exemplu) se realizează cu ajutorul fișierului de configurație `/etc/nsswitch.conf` . Sintaxa acestui fișier este următoarea:

```
bază_de_date_1: sursă_1_1 sursă_1_2 ...
bază_de_date_2: sursă_2_1 sursă_2_2 ...
...
```

Câmpul `bază_de_date` poate fi `hosts` pentru rezolvarea numelor stațiilor, `passwd` pentru rezolvarea numelor de utilizatori, `group` pentru rezolvarea numelor grupurilor de utilizatori, etc. Câmpurile *sursă_1_1*, *sursă_1_2* specifică ordinea metodelor folosite pentru rezolvarea numelor. Cele mai folosite metode sunt:

- `files` pentru a folosi fișierele de configurație din directorul `/etc` (e.g. `/etc/hosts` conține o listă de corespondențe statice între nume de stații și adrese IP)
- `DNS` pentru a folosi serverul de nume specificat în fișierul `/etc/resolv.conf`
- `NIS` pentru a folosi protocolul de administrare centralizată NIS (*Network Information Service).*

7.2.2 Utilitare de interogare DNS

În continuarea acestui capitol se va prezenta configurarea unui server DNS pe Linux. Datorită complexității sistemului DNS, inevitabil vor apărea probleme cauzate de o greșeală de configurare, de configurații neinspirate sau chiar de convergența lentă a protocolului. De aceea se vor prezenta mai întâi moduri de interogare și verificare a serviciului.

În lumea UNIX există trei utilitare mai des folosite: `nslookup`, `host` și `dig`. Cel mai vechi dintre ele, `nslookup`, are un echivalent cu același nume pe platformele Windows. În cele ce urmează va fi prezentat doar utilitarului `host`. Utilitarul `nslookup` este considerat învechit, iar `dig` are o sintaxă mai complicată și un output mai greu de înțeles.

Material cu drept de autor

7.2.2.1 Utilitarul host

Sintaxa de utilizare a comenzii `host` este următoarea:

```
host [-v] [-t tip] [-r] [-l] nume [server]
```

Comanda va încerca să rezolve numele `nume` folosind fie serverul `server` dacă acesta este prezent în lina de comandă, fie serverul implicit configurat în `/etc/resolv.conf` în caz contrar.

Opţiuni ale comenzii sunt:

- `-t` Tipul înregistrării folosit la interogare poate fi configurat cu ajutorul opţiunii `-t`, şi poate fi unul dintre acronimele definite de standardul DNS (`A`, `PTR`, `NS`, etc.) sau `ANY` pentru a întoarce toate înregistrările asociate cu cheia de căutare (numele), indiferent de tipul înregistrării. Dacă nu se foloseşte opţiunea `-t`, atunci `host` va folosi în mod implicit fie `A`, fie `PTR` în funcţie de numele de interogat: dacă numele seamănă cu o adresă IP, se va folosi `PTR`, altfel `A`;
- `-r` În mod implicit interogările realizate de `host` sunt interogări recursive. Pentru interogări nerecursive, trebuie folosită opţiunea `-r`;
- `-l` Uneori poate fi utilă afişarea tuturor intrărilor dintr-o anumită zonă. Cu `host` acest lucru se poate face utilizând opţiunea `-l`. Practic, la folosirea acestei opţiuni, `host` va încerca să facă un transfer de zonă. În funcţie de configurarea serverului, această operaţie poate fi sau nu permisă. Dacă transferul de zonă a fost efectuat, `host` va filtra apoi înregistrările în funcţie de tipul de înregistrare specificat în linia de comandă. Dacă nu se specifică niciun tip de înregistrare în linia de comandă, `host` va lista înregistrările de tip `A` şi `PTR`;
- `-v` Opţiunea `-v` (*verbose*) va afişa odată cu informaţiile cerute şi alte informaţii recepţionate în răspunsul primit de la server.

7.2.3 Configurarea serverului DNS – BIND9

Una dintre primele implementări ale unui server de DNS a fost făcută de către Universitatea Berkeley din California. **BIND** (*Berkeley Internet Name Server Daemon*) este de departe cea mai răspândită implementare de server DNS. Pentru descrierea serverului se va folosi versiunea 9. Aceasta este cea mai recentă dintre versiunile stabile.

7.2.3.1 Instalarea serverului

Petru a instala BIND pe sistemele Ubuntu/Debian se foloseşte utilitarul `apt`.

```
waters@myr:~$ sudo apt-get install bind9
```

Odată instalat, serverul DNS îşi va pune toate fişierele de configurare în `/etc/bind`.

7.2.3.2 Pornirea, oprirea şi restartea serverului

Orice server instalat pe un sistem Ubuntu/Debian îşi va instala un script de iniţializare în `/etc/init.d/`. În cazul BIND, acest script poartă numele de **bind9**. Acesta poate primi ca parametrii: start, stop, restart.

```
waters@myr:/home/rl# /etc/init.d/bind9 stop
 * Stopping domain name service... bind                              [ OK ]

waters@myr:/home/rl# /etc/init.d/bind9 start
 * Starting domain name service... bind                              [ OK ]

waters@myr:/home/rl# /etc/init.d/bind9 restart
 * Stopping domain name service... bind                              [ OK ]
 * Starting domain name service... bind                              [ OK ]
```

7.2.3.3 Fişierul principal de configurare

Fişierul principal de configuraţie este `/etc/bind/named.conf`. Acesta cuprinde două părţi: o primă parte de **opţiuni** şi o a doua de definire de **zone** (domenii). De asemenea,

Material cu drept de autor

înaintea părții de opțiuni poate fi precizat un fișier al cărui conținut va fi inclus în fișierul de configurare prin folosirea unei directive de tipul:

```
include fisier;
```

Pentru o organizare mai bună se recomandă ca opțiunile serverului să fie completate în fișierul `/etc/bind/named.conf.options`, iar definirea de zone să se facă în fișierul `/etc/bind/named.conf.local`. Fișierul `/etc/bind/named.conf` va conține doar opțiunile globale și declarația de zonă implicită (`localhost`) și va include cele două fișiere anterioare. De fapt, cele 2 fișiere sunt incluse în mod implicit de la instalarea serverului:

```
include "/etc/bind/named.conf.options";
// zone default - sintaxa suportă comentarii C,C++,shell
include "/etc/bind/named.conf.local";
```

Atât sintaxa DNS cât și opțiunile puse la dispoziție sunt destul de greu de înțeles în afara unui context practic și de aceea, în continuare se vor prezenta configurații începând de la cele mai simple și des întâlnite (server *caching-only*, server autoritar pe un singur domeniu) până la configurații avansate (*split-brain* DNS, *master/slave*, ACL-uri, delegări de subdomenii). Pe exemplele oferite se va releva importanța fiecărei opțiuni și se va discuta și sintaxa folosită pentru definirea zonelor (domeniilor).

7.2.3.4 Configurarea unui server caching-only

Odată instalat, serverul BIND va porni printr-un script de inițializare. Deși inițial nu a fost configurat niciun domeniu (sau zonă, în terminologia BIND), serverul poate îndeplini rolul de *caching-only* server în mod implicit. Acest lucru este posibil deoarece serverul de nume are în fișierul `db.root`, adresele mai multor servere rădăcină pe care le poate întreba de practic orice domeniu. Serverele rădăcină sunt grupate în domeniul `root-servers.net` și sunt numite A, B, C, D, etc.

```
waters@myr:~$ cat /etc/bind/db.root
[...]
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
A.ROOT-SERVERS.NET.      3600000      AAAA  2001:503:BA3E::2:30
;
; formerly NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     192.228.79.201
;
; formerly C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
[...]
```

7.2.3.4.1 Configurarea unui server forwarder

În acest moment interogarea DNS durează destul de mult pentru numele de domeniu ce nu se află în *cache*. Pentru a putea obține performanțe mai bune se pot configura unul sau mai multe servere cu rol de *forwarder*. În fișierul `/etc/bind/named.conf.options` de mai jos s-a configurat ca server *forwarder*, IP-ul serverul de DNS oferit de ISP.

```
options {
        directory "/var/cache/bind";
        [..]
        forwarders {
              141.85.37.11;
         };
};
```

Material cu drept de autor

Directiva **directory** stabilește directorul de lucru. Din acest director sunt încărcate fișierele indicate în secțiunea de definire de zone dacă nu sunt precizate prin căi absolute. Tot în acest director serverul creează fișiere temporare și poate crea fișiere ce conțin intrările din *cache*.

Directiva **forwarders** permite introducerea unor adrese IP care vor fi folosite ca servere *forwarder* ce vor fi întrebate mereu primele pentru o traducere ce nu s-a putut efectua local.

> ***Atenție!*** *Sintaxa acestei directive, ca și a multor altora din configurația BIND, este destul de rigidă.*

7.2.3.4.2 Configurarea de ACL-uri. Directiva allow-query.

Singura problemă este ca serverul până acum configurat răspunde la cereri recursive efectuate de orice `host` din Internet care poate accesa serverul. Nefiind autoritar peste niciun domeniu, acest server nu are niciun motiv pentru a fi interogat din afara rețelei locale. Într-un scenariu real, configurația unui server *caching-only* va arăta astfel:

```
acl LAN {
                192.168.0.0/24;
        };
options {
        directory "/var/cache/bind";
        allow-query { LAN; };
        […]

         forwarders {
                141.85.37.11;
         };
};
```

La fișierul `/etc/bind/named.conf.options`, de mai sus, s-au adăugat 2 componente:

- s-a definit un **acl** (*access control list*) DNS foarte simplu cu numele de LAN. Directiva `acl` realizează o asociere între numele „LAN" și rețeaua 192.168.0.0/24.
- s-a folosit directiva **allow-query** prin care s-au precizat clienții ce pot trimite cereri serverului de nume. Argumentul din acolade, oferit directivei, ar fi putut fi direct adresa de rețea, însă se preferă folosirea unui *acl* pentru scalabilitate: un `acl` poate avea o structură destul de complexă, permițând unele subrețele și negând accesul altora.

Concluzionând, configurația de mai sus:

- permite cereri doar din partea IP-urilor din rețeaua 192.168.0.0/24;
- folosește ca *forwarder* serverul DNS cu adresa 141.85.37.11;
- stochează fișierele pentru *cache* în `/var/cache/bind`.

7.2.3.5 Configurarea unui server autoritar peste un domeniu

În general un server de nume autoritar știe să facă atât operația de rezolvare directă cât și operația de rezolvare inversă pentru un domeniu pe care îl deține. Pentru fiecare tip de rezolvare este nevoie de definirea unei **zone.**

> *În terminologia DNS, o zonă reprezintă un mod de a grupa informația de care este nevoie pentru a face una dintre rezolvări: directă sau inversă.*

Deci daca se dorește realizarea rezolvării directe și inverse, trebuie definite 2 zone, pentru fiecare domeniu. După cum s-a specificat, pentru rezolvarea inversă va trebui creat un subdomeniu de tipul: **IP.in-addr.arpa**.

Pentru exemplificare se va presupune crearea zonei `politehnica.ro`, alături de zona `in-addr.arpa` corespunzătoare. Specificațiile pe care domeniul va trebui să le respecte, sunt după cum urmează:

- serverul de nume pentru domeniul `politehnica.ro` este `ns.politehnica.ro` și are adresa IP `142.100.111.53`
- serverul de nume `ns.politehnica.ro` va răspunde la cereri pentru înregistrările:

Material cu drept de autor

- www.politehnica.ro – 142.100.111.80
- ftp.politehnica.ro – 142.100.111.21
- mail.politehnica.ro – 142.100.111.25

- serverul va oferi posibilitatea de rezolvare inversă din adrese IP în nume

Operația de a crea o zonă pentru care serverul de nume să fie autoritar, constă din 2 pași:

- **Definirea zonei** în fișierul `/etc/bind/named.conf.local` sau în fișierul principal de configurare.
- Introducerea **parametrilor zonei** și rezolvările pentru diferite adrese, într-un **fișier de zonă** (fișier de configurare pentru respectiva zonă).

7.2.3.5.1 Definirea zonei

Fișierul `/etc/bind/named.conf.local` va trebui sa conțină următoarele informații despre zona:

- Numele zonei
- Calea în sistemul de fișiere către fișierul de configurare al zonei
- Tipul zonei (master/slave)

O definire corectă pentru cele 2 zone ale domeniului `politehnica.ro`, arată astfel:

```
waters@myr:~$ cat /etc/bind/named.conf.local

zone "politehnica.ro" {
        type master;
        file "/etc/bind/db.politehnica.ro";
};

zone "111.100.142.in-addr.arpa" {
        type master;
        file "/etc/bind/db.111.100.142.in-addr.arpa";
};
```

Se va analiza în continuare sintaxa și funcționalitatea fișierului de mai sus.

Pentru a putea specifica o zona se folosește directiva **zone** urmată de numele efectiv al zonei, plasat între ghilimele, tipul și calea către fișierul de configurare al zonei dată în format absolut. Dacă în locul căii absolute s-ar fi specificat numele în mod relativ: `db.politehnica.ro` și respectiv `db.111.100.142.in-addr.arpa`, serverul DNS ar fi concatenat numele în mod implicit la calea referită de directiva **directory**. Cum aceasta este setată implicit la adresa `/var/cache/bind/`, adresa finală ar fi fost `/var/cache/bind/db.politehnica.ro`.

> ***Notă:*** *nu este obligatoriu ca numele fișierelor de zonă să respecte convenția de nume* ***db.nume_domeniu****, însă modul acesta de denumire a fost adoptat de majoritatea administratorilor pentru a oferi un mod clar de a recunoaște fișierul de zonă al fiecarui domeniu.*

> ***Atenție!*** *Sintaxa de definire a zonei este destul de strictă și trebuie realizată urmărind exemplul de mai sus.*

7.2.3.5.2 Fișiere de zonă

După cum se observă din sintaxa fișierului principal de configurație, fiecare domeniu sau zonă are asociat un fișier de configurație. Acesta conține practic baza de date DNS pentru domeniu. Fișierul conține mai multe înregistrări DNS, fiecare înregistrare fiind descrisă pe un singur rând. Excepție fac înregistrările de tip `SOA` care se încheie odată cu caracterul „`)`”.

Material cu drept de autor

Înregistrările sunt structurate la rândul lor în câmpuri separate de spaţii sau tab-uri. Formal, structura fişierelor de zonă este următoarea:

```
$INCLUDE fişier nume_domeniu          //
$ORIGIN nume_domeniu                  // parametrii de zonă
$TTL ttl                              //

Nume_domeniu ttl clasa tip informaţii      // înregistrare DNS
```

Dacă numele unui domeniu nu se termină cu . atunci este considerat **nume incomplet**. Pentru a afla numele complet folosit în înregistrări se concatenează cu `domeniu` `nume_domeniu` definit de operatorul `$ORIGIN`.

O greşeală frecventă atunci când se editează fişierele de configuraţie pentru zone este scrierea incorectă a numelor complete, prin uitarea aplicării punctului la sfârşitul numelui.

Operatorul `$INCLUDE` specifică fişierul ce va fi inclus înainte de parsare. Dacă este prezent şi parametrul `nume_domeniu`, acesta va preciza numele concatenat la numele incomplete, dar doar pentru înregistrările din fişierul inclus.

Operatorul `$ORIGIN` precizează numele ce se va concatena la numele incomplete din secţiunea ce urmează.

Operatorul `$TTL` indică timpul maxim pentru care un răspuns pozitiv pentru intrări din secţiunea curentă poate sta în *cache*-ul altor servere.

7.2.3.5.3 Configurarea fişierului de zonă pentru rezolvare directă

Pentru o mai bună înţelegere a parametrilor de mai sus şi a sintaxei înregistrărilor DNS, se va analiza modul în care ar trebui completat fişierul de zonă `db.politehnica.ro`.

```
$ORIGIN ro.
$TTL 36000
politehnica    IN SOA ns.politehnica.ro. admin.politehnica.ro. (
                      2007092001  ; Serial
                      8H          ; Refresh
                      2H          ; Retry
                      1W          ; Expire
                      1D          ; Negative TTL
                      )
politehnica           TXT         "Universitatea Plitehnica Bucuresti"
politehnica           NS          ns.politehnica
politehnica           MX    10    mail.politehnica
politehnica           A           142.100.111.53

ns.politehnica        IN    A     142.100.111.53
mail.politehnica      IN    A     142.100.111.25
www.politehnica.ro.   IN    A     142.100.111.80
ftp.politehnica.ro.   IN    A     142.100.111.21
```

Parametrul `$ORIGIN` va avea ca efect concatenarea domeniului `ro`. Tuturor domeniilor din fişier ce nu se termină cu caracterul . (punct). Spre exemplu numele `ns.politehnica` se va transforma după parsare în `ns.politehnica.ro`.

Din punct de vedere funcţional, fişierul de configurare de mai sus s-ar traduce astfel:

- Domeniul peste care serverul este autoritar este `politehnica.ro` (în fişier este completat doar cuvântul `politehnica`, însă acestuia îi este concatenat parametrul `$ORIGIN`)
- Clasa tuturor înregistrărilor este IN. Aceasta este clasa folosită peste tot în Internet. Introducerea ei este opţională.
- Domeniul `politehnica.ro` are 5 înregistrări
  - SOA – acest tip de înregistrare are o sintaxa diferită de cea standard şi va di discutată mai jos.
  - TXT – înregistrarea TXT oferă un comentariu legat de societatea ce deţine domeniul
  - NS – această înregistrare precizează serverul de nume care va răspunde la cererile pentru acest domeniu: ns.politehnica.ro (.ro este concatenat de la parametrul $ORIGIN)
  - MX - înregistrarea de tip `MX` are un format special: o prioritate şi apoi adresa serverului. Prioritatea defineşte ordinea în care un client va încerca să contacteze serverele de mail ale domeniului; se va încerca mai întâi conectarea la serverele cu prioritate mai mică.

Material cu drept de autor

- A – face posibilă traducerea acestui domeniu oferind adresa IP pentru `politehnica.ro`

- Pentru domeniul politehnica.ro s-a completat numele unui server de e-mail (intrarea MX) şi serverul de nume (intrarea NS). Fiecare dintre acestea, pentru a putea fi rezolvate, are nevoie de asocierea cu o adresa IP. Acest lucru este realizat de următoarele două înregistrări DNS de tip A.
- De asemenea mai există înca 2 înregistrări A pentru traducerea domeniilor `www.politehnica.ro` şi `ftp.politehnica.ro`. A se observa faptul că cele două nume de domenii au fost date cu nume complet şi încheiate cu caracterul . (punct). Notaţia aceasta este perfect echivalenta cu a scrie: `www.politehnica` şi `ftp.politehnica`, având parametrul `$ORIGIN „ro. "` Trebuie avut atenţie totuşi la folosirea ambelor notaţii, căci se poate întampla ca în loc de `www.politehnica.ro.` , să se completeze doar `www.politehnica.ro` şi să se uite caracterul „.", de la sfârşit. Dacă se configurează astfel, domeniul final va fi `www.politehnica.ro.ro`, şi deci, eronat.
- Ca o ultimă observaţie, comentariile din fişier care au fost făcute cu ajutorul caracterului „;"

Inregistrarea de de tip SOA are o sintaxă mai specială şi trebuie să fie prima înregistrare în fişierul de zonă.

```
domeniu          IN      SOA     server_autoritar adresă_mail_admin (
                                 serial
                                 refresh
                                 retry
                                 expire
                                 ttl
                                 )
```

- *domeniu* = `politehnica.ro.`: Specifică numele complet al domeniului gestionat.
- *server_autoritar* = `ns.politehnica.ro.`: Precizează un server de nume ce va fi trimis drept răspuns atunci când nu se poate răspunde la cerere, dar numele interogat este inclus în domeniul pentru care serverul este autoritar. Aceasta se întâmplă când un server *slave* dă un răspuns negativ. În general, acest câmp trebuie setat la numele serverului *master* pentru domeniul respectiv.
- *adresă_mail_admin* = `admin.politehnica.ro.`: Precizează adresa de e-mail a administratorului domeniului, în care caracterul `@` este înlocuit cu . (ex: în loc de `admin@politehnica.ro`, apare `admin.politehnica.ro.`)
- *serial* = 2007092001: Indică versiunea bazei de date DNS pentru domeniu. Această serie este verificată de serverele *slave* pentru a determina dacă este necesară o descărcare a bazei de date de la *master*. De obicei se foloseşte o convenţie a formatului seriei: `AAAALLZZMM`, unde `AAAA` reprezintă anul, `LL` luna, `DD` ziua, iar `MM` numărul modificării din ziua respectivă. Această convenţie nu este obligatorie, dar este indicată.

Seria bazei de date trebuie incrementată la fiecare modificare făcută în domeniu, altfel modificările nu se vor propaga la serverele slave.

- *refresh* = 8H: Conţine intervalul de timp la care serverul *slave* va interoga seria bazei de date a serverului *master*. Implicit acest timp este exprimat în secunde, dar valoarea câmpului *refresh* poate exprima minutele - dacă se foloseşte sufixul `M`, orele - dacă se foloseşte sufixul `H`, sau zilele - dacă se foloseşte sufixul `D`.
- *retry = 2H:* Indică intervalul de timp la care serverul *slave* va încerca să se reconecteze la serverul *master*, dacă o încercare de conectare la server a eşuat.
- *expire = 1W:* Indică perioada de timp după care baza de date a serverului *slave* este invalidată, dacă nu se reuşeşte o conectare la serverul *master*.
- *ttl = 1D:* Reprezintă timpul de viaţă pentru răspunsuri negative. Acest timp este diferit de timpul de viaţa al răspunsurilor pozitive.

Material cu drept de autor

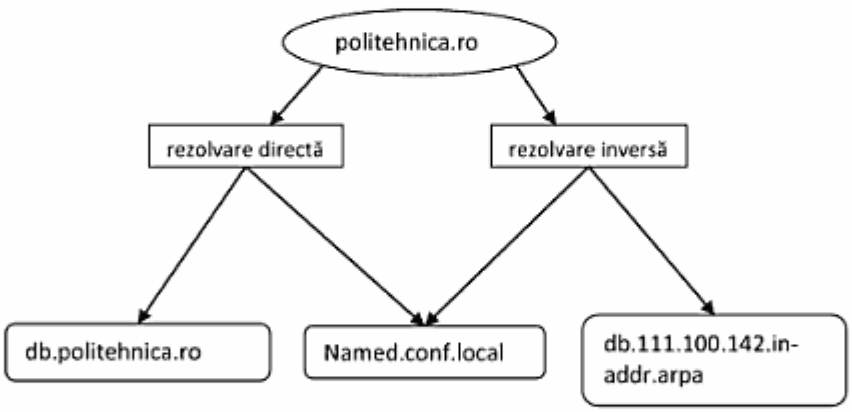
7.2.3.5.4 Configurarea fişierului de zonă pentru rezolvarea inversă

În acest moment serverul DNS a fost configurat pentru rezolvare directă. În continuare se va exemplifica şi fişierul de zonă pentru rezolvarea inversă: `db.111.100.142.in-addr.arpa`

```
$ORIGIN 100.142.in-addr.arpa.
111     IN      SOA     ns.politehnica.ro. nsmaster.politehnica.ro. (
                        2007092001      ; Serial
                        8H              ; Refresh
                        2H              ; Retry
                        1W              ; Expire
                        1D              ; Negative TTL
                        )
                TXT     "Universitatea Politehnica Bucuresti"
                NS      ns.politehnica.ro.

$ORIGIN 111.100.142.in-addr.arpa.

21                      PTR     ftp.politehnica.ro.
25                      PTR     mail.politehnica.ro.
80                      PTR     www.politehnica.ro.
```

Exceptând înregistrarea `SOA`, pentru domeniul `111.100.142.in-addr.arpa` se adaugă o înregistrare text de descriere a domeniului, una pentru serverul de nume şi alte 3 înregistrări de tip pointer pentru rezolvarea `21.111.100.142.in-addr.arpa` în `ftp.politehnica.ro`, `25.111.100.142.in-addr.arpa` în `mail.politehnica.ro` şi `80.111.100.142.in-addr.arpa` în `www.politehnica.ro`.

În schema de mai jos sunt prezentate fişierele create/editate/descrise în secţiunea de mai sus şi modul în care fiecare contribuie la traducerea directă şi inversă a domeniului `politehnica.ro`



7-3: Fişiere de configurare

7.2.3.6 Verificarea serverului DNS

Se recomandă urmărirea exemplului de mai sus şi implementarea acestuia pe o staţie locală, fiind o configuraţie simplă de server DNS, care va constitui baza pentru configuraţii mai avansate.

7.2.3.7 Verificarea sintaxei

Din păcate, în realizarea primelor configurări de DNS se poate greşi foarte uşor la sintaxa destul de strictă a fişierelor de configurare. De aceea serverul BIND pune la dispoziţie două utilitare cu care se poate verifica fişierul principal de configurare (alături de toate fişierele incluse în acesta) şi fişierele de zonă. Acestea se numesc: `named-checkconf` şi `named-checkzone`.

Material cu drept de autor

```
waters@myr:/etc/bind# named-checkconf
waters@myr:/etc/bind# named-checkzone politehnica.ro. /etc/bind/db.politehnica.ro
zone politehnica.ro/IN: loaded serial 2007092001
OK

waters@myr:/etc/bind#              named-checkzone              111.100.142.in-addr.arpa
/etc/bind/db.111.100.142.in-addr.arpa
zone 111.100.142.in-addr.arpa/IN: loaded serial 2007092001
OK
```

7.2.3.8 Testarea funcționării

Pentru a putea testa buna funcționare a domeniului `politehnica.ro`, va trebui să se restarteze serviciul bind9 şi să se interogheze serverul cu utilitarul `host`.

```
root@myr:/etc/bind# /etc/init.d/bind9 restart
 * Stopping domain name service... bind                    [ OK ]
 * Starting domain name service... bind                    [ OK ]

user@orange:/etc/bind# host www.politehnica.ro 142.100.111.53
Using domain server:
Name: 142.100.111.53
Address: 142.100.111.53#53
Aliases:

www.politehnica.ro has address 142.100.111.80

user@orange:/etc/bind# host -t PTR 80.111.100.142.in-addr.arpa
80.111.100.142.in-addr.arpa domain name pointer www.politehnica.ro.
```

La apelarea utilitarului `host` s-a folosit şi argumentul 142.100.111.53 prin care s-a specificat adresa serverului DNS pe care host va trebui să îl interogheze; bineînțeles acea adresa IP este adresa IP a serverului pe care s-au făcut configurările de mai sus. Se observă că cele 2 interogări efectuate cu utilitarul `host` sunt rezolvate cu succes de către serverul DNS. Este vorba de rezolvarea directă a adresei `www.politehnica.ro` în `142.100.111.80` şi de rezolvarea inversă a `142.100.111.80` în `www.politehnica.ro`.

7.2.3.9 BIND9 debugging

Deşi BIND9 include cele două utilitare, `named-checkconf` pentru verificarea fişierului principal de configurare şi `named-checkzone` pentru verificarea zonelor, acestea nu oferă întotdeauna cele mai intuitive mesaje de eroare. Din acest motiv, în continuare se vor prezenta scenarii de *debugging* mai încăpățânate şi sursa greşelilor din acestea.

10. Eroarea numelor

Cea mai întâlnită greşeală este fără îndoiala uitarea punctului. Cea mai bună indicatie asupra acestui fapt este eroarea: **ignoring out-of-zone data**. Aceasta poate însemnă totuşi şi faptul că numele de domeniu este scris greşit în SOA sau că zona are numele scris greşit în definirea acesteia din fisierul `named.conf.local`.

11. Eroarea copy-paste

Având în vedere sintaxa complicată BIND, se recomandă păstrarea unor *template*-uri care să fie folosite la configurații noi. Însă trebuie foarte multă atenție la copierea dintr-un fişier în altul deoarece simpla copiere a unui spațiu în plus, luat odată cu numele de domeniu, poate să mute caracterul „.” mai la dreapta şi să rezulte în această eroare: **not a valid number**.

12. Eroarea neraportată

Una din cele mai greu de depistat erori constă în **a scrie greşit numele de cale** al fişierului de zonă în fişierului `named.conf.local`. Deşi serverul DNS nu va funcționa pentru respectiva zonă, acesta va porni fără să ofere o eroare. De asemenea niciunul din utilitarele `named-check` nu raportează ceva în neregulă.

Material cu drept de autor

Pentru alte greşeli comune şi puncte cheie care presupun o atenţie sporită, se poate consulta RFC1912[1] care tratează tocmai aceste subiecte.

7.2.3.10 Eficientizarea sintaxei fişierelor de zonă

Deşi fişierul de zonă configurat mai sus este corect specificat, există unele mici artificii de sintaxă care sunt întâlnite destul de des în implementarea unui server DNS şi care trebuie ştiute de orice administrator de BIND.

1. Omiterea numelui de domeniu ce se repetă consecutiv.

Pentru început se poate observa că în fişierul `db.politehnica.ro`, dat ca exemplu mai sus, domeniul `politehnica.ro` a avut 5 înregistrări diferite: SOA, TXT, MX, NS, A. Sintaxa DNS generală pentru o înregistrare este:

```
nume_domeniu ttl clasa tip informatii      // înregistrare DNS
```

Cuvântul „politehnica" a apărut deci de 5 ori ca nume de domeniu. Acest lucru nu este neapărat necesar. Atunci când numele de domeniu este la fel pentru înregistrările ce urmează, acesta poate fi omis:

```
[...]
politehnica   IN SOA ns.politehnica.ro. admin.politehnica.ro. (
                     2007092001  ; Serial
                     8H          ; Refresh
                     2H          ; Retry
                     1W          ; Expire
                     1D          ; Negative TTL
                     )
              TXT            "Universitatea Plitehnica Bucuresti"
              NS             ns.politehnica
              MX     10      mail.politehnica
              A              142.100.111.53

ns.politehnica       IN     A       142.100.111.53
[...]
```

2. Simbolul „@"

Există multe exemple de configuraţii BIND pe Internet. Multe din acestea conţin în fişierul de zona simbolul „@". Simbolul acesta este echivalent cu argumentul directivei `$ORIGIN`. În exemplul de mai jos, a scrie „politehnica.ro." sau a scrie „@" ar avea acelaşi efect. Deci un alt mod de a scrie fişierul `db.politehnica.ro` este următorul:

```
$ORIGIN politehnica.ro.
$TTL 36000
@   IN SOA ns.politehnica.ro. admin.politehnica.ro. (
                     2007092001  ; Serial
                     8H          ; Refresh
                     2H          ; Retry
                     1W          ; Expire
                     1D          ; Negative TTL
                     )
              TXT            "Universitatea Plitehnica Bucuresti"
              NS             ns
              MX     10      mail
              A              142.100.111.53

ns      IN      A       142.100.111.53
mail    IN      A       142.100.111.25
www     IN      A       142.100.111.80
ftp     IN      A       142.100.111.21
```

Se observă că argumentul `$ORIGIN` a fost schimbat din „ro." în „politehnica.ro.", iar în locul numelui de domeniu din faţa înregistrării SOA, s-a introdus simbolul „@". Acesta este perfect echivalent cu a scrie `politehnica.ro.`

[1] http://www.ietf.org/rfc/rfc1912.txt

Material cu drept de autor

În continuare, la înregistrările TXT, NS, MX şi A, nu s-a mai introdus simbolul „@", conform regulii de la punctul 1. De asemenea cu cât directiva `$ORIGIN` este mai cuprinzătoare cu atât fişierul mai aerisit şi citirea sa mai uşoară.

3. Redefinirea parametrului `$ORIGIN`

Sintaxa DNS permite redefinirea parametrului `$ORIGIN` pentru o mai mare flexibilitate a fişierelor de zonă.

```
$ORIGIN ro.
$TTL 36000
politehnica   IN SOA ns.politehnica.ro. admin.politehnica.ro. (
                     2007092001  ; Serial
                     8H          ; Refresh
                     2H          ; Retry
                     1W          ; Expire
                     1D          ; Negative TTL
                     )
              TXT          "Universitatea Plitehnica Bucuresti"
              NS           ns.politehnica
              MX     10    mail.politehnica
              A           142.100.111.53
$ORIGIN politehnica.ro.

ns      IN      A       142.100.111.53
mail    IN      A       142.100.111.25
www     IN      A       142.100.111.80
ftp     IN      A       142.100.111.21
```

În fişierul de mai sus, pentru primele 5 înregistrări, numele care nu se termină cu caracterul „.", vor fi completate cu „`ro.`". Pentru următoarele 3 înregistrări parametrul $ORIGIN a fost schimbat, deci numele se vor completa cu „`politehnica.ro.`"

7.2.3.11 Configurarea unui alias DNS

Pentru a configura un alias DNS se foloseşte înregistrarea CNAME. Un nume canonic nu poate fi configurat decât pentru un subdomeniu al domeniului peste care serverul este autoritar. În caz contrat oricine ar putea realiza un alias pentru `google.com` care să se traducă într-un IP dorit. Configurarea oferită ca exemplu ca crea `f1.politehnica.ro` ca alias pentru `www.politehnica.ro`.

```
$ORIGIN politehnica.ro.
[...]

www     IN      A       142.100.111.80
f1      IN      CNAME   www

[...]
```

După cum se poate vedea şi din interogare, `f1.politehnica.ro` va fi tradus în acelaşi IP ca şi `www.politehnica.ro`.

```
root@myr:/etc/bind# host www.politehnica.ro 142.100.111.53

Using domain server:
Name: 142.100.111.53
Address: 142.100.111.53#53
Aliases:

www.politehnica.ro has address 142.100.111.80

user@orange:/etc/bind# host f1.politehnica.ro 142.100.111.53

Using domain server:
Name: 142.100.111.53
Address: 142.100.111.53#53
Aliases:

www.politehnica.ro has address 142.100.111.80
```

Verificarea cu argumentul `-t CNAME` oferă direct rezultatul alias-ului:

```
waters@myr:~$ host -t CNAME f1.politehnica.ro 142.100.111.80
Using domain server:
Name: 142.100.111.80
Address: 142.100.111.80#53
```

Material cu drept de autor

```
Aliases:

f1.politehnica.ro is an alias for www.politehnica.ro.
```

7.3 Configurări avansate DNS

7.3.1 Delegarea unui subdomeniu DNS.

Delegarea DNS este un proces ce a stat la baza întregii ierarhii a protocolului. Pentru a analiza acest concept se va considera o situație în care un administrator de rețea dorește să își cumpere un domeniu de la ROTLD (*Romanian Top Level Domain*), cu numele: `poliadmin.ro`. ROTLD, care deține domeniul „`.ro`", va trebui să delege autoritatea pentru subdomeniul pe care îl vinde. Mai exact, autoritatea ROTLD va crea de acum o asociere între IP-ul pe care administratorul îl va asigna stației ce găzduiește serverul de nume și domeniul `poliadmin.ro`. Această asociere va putea fi comunicată de către TLD-ul „`.ro`", oricărui server din Internet care va dori să o afle.

Mai mult, administratorul care a cumpărat domeniul `poliadmin.ro`, poate delega mai departe subdomenii din acesta: `foxxy.poliadmin.ro`, `suzy.poliadmin.ro`, etc. Când serverul de nume autoritar peste `poliadmin.ro` va primi o cerere nerecursivă pentru domeniul `suzy.poliadmin.ro`, va răspunde cu adresa IP a serverului de nume căruia a delegat autoritatea pentru respectivul subdomeniul.

Singura înregistrare cu care se poate realiza delegarea de domenii este înregistrarea NS. Pentru a clarifica mai bine acest concept se va continua exemplul de mai sus cu delegarea subdomeniului `cs.politehnica.ro`. Răspunzător de acest subdomeniu este serverul de nume `ns.cs.politehnica.ro`. Adresa acestuia (`142.100.222.53`) va fi specificată printr-o înregistrare de tip `A`.

Procesul de delegare are loc de fapt în fișierul de zonă `db.politehnica.ro`. Fisierul va arăta astfel:

```
$ORIGIN politehnica.ro.
$TTL 36000
@   IN SOA ns.politehnica.ro. admin.politehnica.ro. (
                        2007092001  ; Serial
                        8H          ; Refresh
                        2H          ; Retry
                        1W          ; Expire
                        1D          ; Negative TTL
                        )
                TXT             "Universitatea Pltehnica Bucuresti"
                NS              ns
                MX      10      mail
                A               142.100.111.53

ns      IN      A       142.100.111.53
mail    IN      A       142.100.111.25
www     IN      A       142.100.111.80
ftp     IN      A       142.100.111.21

; urmează partea de delegare

cs      NS      ns.cs
ns.cs   A       142.100.222.53
```

Ultimele 2 linii din fișier reprezintă delegarea domeniului `cs.politehnica.ro` către serverului de nume `ns.cs.politehnica.ro`, de pe stația cu IP-ul 142.100.22.53. Înregistrarea de tip NS specifică serverul de nume pentru numele `cs.politehnica.ro`, iar înregistrarea de tip A asociaza serverul de nume cu un IP. Acum autoritatea a fost delegată. Nu mai rămâne decât să se configureze serverul pentru `cs.politehnica.ro`, pentru a avea cine să răspundă la o cerere pentru acest domeniu.

Configurarea decurge în mod asemănător cu cea pentru serverul autoritar domeniului `politehnica.ro`.

Material cu drept de autor

În fişierul de zone se adaugă o singură intrare de zonă, cea corespunzătoare rezolvării directe a domeniului `cs.politehnica.ro`:

```
zone "cs.politehnica.ro" {
        type master;
        file "/etc/bind/db.cs.politehnica.ro";
};
```

Pentru fişierul de zonă `db.cs.politehnica.ro` se adaugă aceleaşi tipuri de înregistrări ca în cazul `db.politehnica.ro`, cu exceptia înregistrărilor corespunzătoare delegării de subdomeniu.

```
$ORIGIN politehnica.ro.
$TTL 36000
cs IN SOA ns.cs.politehnica.ro. admin.cs.politehnica.ro. (
                        2007092001      ; Serial
                        8H              ; Refresh
                        2H              ; Retry
                        1W              ; Expire
                        1D              ; Negative TTL
                        )
                        TXT             "Computer Science Department"
                        NS              ns.cs
                        MX      10      mail.cs
                        A               142.100.222.53

$ORIGIN cs.politehnica.ro.

ns                      A       142.100.222.53
www                     A       142.100.222.80
ftp                     A       142.100.222.21
mail                    A       142.100.222.25
```

Se observă similaritatea foarte mare a configuraţiei fişierului de zonă `db.cs.politehnica.ro` cu fişierul `db.politehnica.ro`. Excepţiile constau în numele de domeniu şi adresele IP.

7.3.2 Efectuarea DNS load-balancing.

Cu ajutorul serverului DNS se poate implementa o formă destul de primitivă de *load balancing,* în care clientul va primi într-o maniera *round-robin*[1] un IP dintr-o listă de n IP-uri. Astfel dacă n clienţi vor face câte o cerere fiecare, vor primi n IP-uri diferite. Implementarea efectivă este destul de simplă în serverul BIND şi presupune existenţa a mai multe înregistrări de tip A pentru acelaşi nume de domeniu.

```
[...]
www     IN      A       142.100.111.80
        IN      A       142.100.111.81
        IN      A       142.100.111.82

ftp     IN      A       142.100.111.21
[...]
```

Se observă cum la realizarea unor cereri cu ajutorul utilitarului `host`, serverul răspunde cu o listă de IP-uri permutata într-o maniera *round-robin*. De obicei un client care va primi 3 răspunsuri de tip A, îl va alege pe primul şi le va ignora pe restul.

```
waters@myr:~$ host www.politehnica.ro 142.100.111.53
Using domain server:
Name: 142.100.111.53
Address: 142.100.111.53#53
Aliases:

www.radiance.ro has address 142.100.111.80
www.radiance.ro has address 142.100.111.81
www.radiance.ro has address 142.100.111.82

waters@myr:~$ host www.radiance.ro 142.100.111.53
Using domain server:
Name: 142.100.111.53
Address: 142.100.111.53#53
Aliases:
```

[1] parcurgere secvenţială a unei liste circulare

Material cu drept de autor

```
www.radiance.ro has address 142.100.111.82
www.radiance.ro has address 142.100.111.80
www.radiance.ro has address 142.100.111.81

waters@myr:~$ host www.radiance.ro 142.100.111.53
Using domain server:
Name: 142.100.111.53
Address: 142.100.111.53#53
Aliases:

www.radiance.ro has address 142.100.111.81
www.radiance.ro has address 142.100.111.82
www.radiance.ro has address 142.100.111.80
```

7.3.3 Configurarea DNS Master/Slave.

Majoritatea serverelor DNS implementate într-un scenariu real au în componenţă şi un server slave aflat pe o maşina fizică separată care să poată interveni în cazul în care masterul ar înceta să mai funcţioneze. La un interval specific, serverul ca copia fişierele de zonă de la master şi astfel se va păstra întotdeauna sincronizat cu acesta. Configuraţia de master slave este prezentată în continuarea exemplului de bază folosit în acest capitol pentru domeniul `politehnica.ro`.

În primul rând, fişierul de zonă va trebui modificat pentru a oferi o traducere şi pentru serverul de nume care este slave. Se vor introduce două înregistrări NS, în loc de una, pentru ambele severe de nume.

```
$ORIGIN politehnica.ro.
$TTL 36000
@   IN SOA ns1.politehnica.ro. admin.politehnica.ro. (
                        2008090501  ; Serial
                        8H          ; Refresh
                        2H          ; Retry
                        1W          ; Expire
                        1D          ; Negative TTL
                        )
                TXT             "Universitatea Plitehnica Bucuresti"
                NS              ns1.politehnica.ro.
                NS              ns2.politehnica.ro.
                MX      10      mail.politehnica.ro.
                A               142.100.111.53

ns1     IN      A       142.100.111.53
ns2     IN      A       142.100.99.1
mail    IN      A       142.100.111.25
www     IN      A       142.100.111.80
ftp     IN      A       142.100.111.21
```

Trebuie făcute două observaţii legate de fişierul de mai sus:

- deşi s-au definit două servere NS, doar masterul este introdus în înregistrarea SOA
- serverul ns1 şi ns2 sunt localizate pe maşini diferite (IP-uri diferite în înregistrările de tip A); redundanţa este dusă chiar mai departe prin faptul ca cele două maşini se află în *subneturi* diferite cu masca /24.

Urmează definirea zonelor pentru fiecare server. Configurările se vor face direct în fişierul principal de configurare (fără a mai folosi `named.conf.options` şi `named.conf.local`)

Serverul *master* va avea următoarea intrare de zonă în `named.conf`:

```
zone "politehnica.ro" {
        type master;
        file "/etc/bind/db.politehnica.ro";
        notify yes;
        also-notify {
                141.85.37.33;
                193.230.31.225;
        };
        allow-transfer {
                142.100.99.1
        };
};
```

Observaţii:

- tipul serverului este setat ca master

Material cu drept de autor

- directiva **notify:** atunci când această directivă este folosită şi setata la valoarea **yes,** la fiecare schimbare a fişierului de zonă se vor trimite mesaje speciale de **NOTIFY** tuturor serverelor *slave* care sunt specificare în fişierul de zonă prin înregistrarea **NS** (adică serverele de nume cu înregistrări NS, în afara de cel master care este specificat în **SOA**).
- directiva **allow-transfer:** permite transferul doar către serverul *slave* cu IP-ul specificat. Dacă aceasta directivă ar lipsi, orice server ar putea să descarce configuraţiile de zonă de la *master*.
- directiva **also-notify**: este folosită pentru a notifica şi alte servere, care se află de obicei mai sus în ierarhia DNS, de schimbarea unui fişier de zonă. Este trimis tot un mesaj de tip NOTIFY, odată cu mesajele trimise şi serverelor *slave*.
- cele 3 opţiuni specificate prin directivele de mai sus se vor aplica doar zonei `politehnica.ro`! Dacă acestea ar fi fost introduse în `named.conf.options` sub directiva globală **options,** acestea s-ar fi aplicat global.

Intrarea de zonă din fişierul de configurare al serverului slave va arăta astfel:

```
zone "politehnica.ro" {
        type slave;
        file "/etc/bind/db_slave.politehnica.ro.txt";
        masters { 142.100.111.53; };
       allow-notify { 142.100.111.53; };
};
```

Observaţii:

- tipul serverului de nume este *slave*
- directiva **masters:** indică *slave*-ului IP-ul serverului *master* de la care să îşi descarce configuraţia
- directiva **allow-notify**: specifică serverele *master* de la care acest *slave* acceptă notificări

7.4 Configurarea unui server DNS pe Windows Server 2008

Buna funcţionare a unui sistem de rezolvare a numelor reprezintă o componentă cheie în orice sistem de operare orientat spre reţea. Însăşi funcţionarea unei reţele se bazează pe posibilitatea ca entităţile ce o formează să se poată localiza între ele. În consecinţă, modul de rezolvare a numelor trebuie să fie flexibil şi robust. De asemenea, este de preferat ca acesta să adere standardelor, din punct de vedere al interoperabilităţii şi al scalabilităţii.

Pentru *Windows Server 2008*, DNS (*Domain Name System*) reprezintă principala modalitate de rezolvare a numelor şi, totodată, reprezintă o componentă de bază a oricărei infrastructuri de tip *Active Directory*.

O alternativă la DNS folosită de unii administratori este WINS (*Windows Internet Name Service*), un serviciu de rezolvare a numelor bazat pe NetBIOS care realizează asocieri între adresele IP ale staţiilor dintr-o reţea şi numele lor NetBIOS, stocându-le într-o bază de date dinamică si distribuită.

> ***NetBIOS*** *(Network Basic Input/Output System) reprezintă un protocol ce rulează peste TCP/IP[1] şi permite calculatoarelor dintr-o reţea să comunice pe baza pe baza unuia sau a mai multor nume de tip NetBIOS. NetBIOS oferă servicii din categoria nivelului sesiune al stivei OSI.*

Atât WINS cât şi DNS reprezintă modalităţi de rezolvare a numelor, disponibile pe sistemele Windows. WINS realizează rezolvarea numelor în spaţiul de nume NetBIOS, în timp ce DNS le rezolvă în spaţiul de nume al domeniilor DNS. În general, WINS este folosit atât de către clienţii ce folosesc versiuni mai vechi de Windows, cât şi de aplicaţiile bazate pe NetBIOS. Windows 2000, XP, 2003 Server şi 2008 Server folosesc şi nume DNS, pe lângă cele oferite de NetBIOS.

[1] NetBIOS peste TCP/IP mai este prescurtat şi NBT.

Material cu drept de autor

7.4.1 Instalare şi configurare

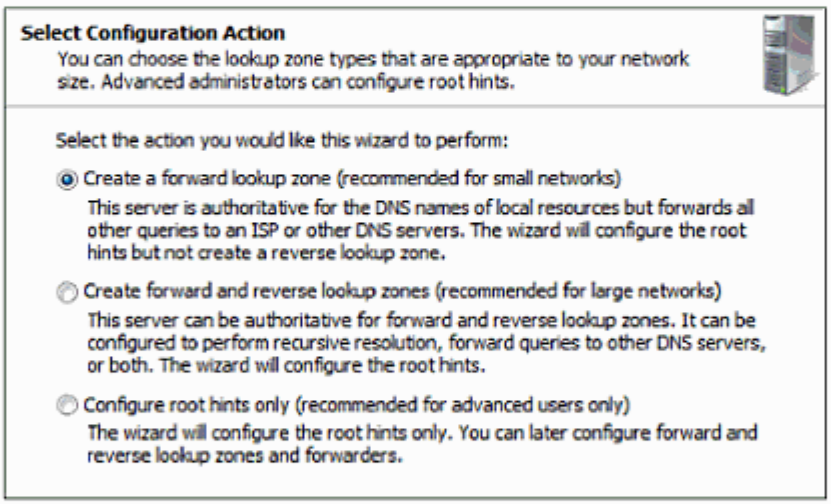
Instalarea serviciului DNS pentru *Windows Server 2008* se realizează prin adăugarea unui rol al serverului. Folosind *Server Manager* se accesează interfaţa *Add Server Roles* şi se selectează DNS Server din listă. Se continuă instalarea, ţinând cont de recomandarea de a avea configurată adresă statică pe interfaţa pe care se doreşte ca serverul DNS să răspundă cererilor. *Server Manager* va atenţiona în cazul în care nu detectează nicio interfaţă configurată static.

DNS se ocupă, în principal, cu zonele de rezolvare directă (*forward lookup zones*), care realizează conversia din nume în adrese IP. De asemenea, el se ocupă şi de conversia inversă, din adrese IP în nume (*reverse lookup*).

7.4.1.1 Configurarea iniţială şi crearea unei zone

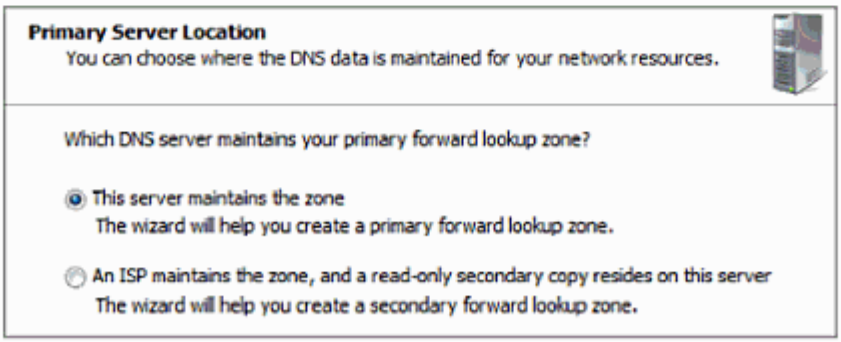
După instalare, interfaţa de administrare a serverului DNS pote fi accesată de la `Start > Administrative Tools > DNS`. Configurarea iniţială se poate realiza de la meniul `Action > Configure a DNS Server`. În etapele ce urmează se vor exemplifica o serie de acţiuni şi configurări pentru a pune în evidenţă principalele facilităţi ale serverului:

1. Pentru început, se alege tipul de zonă ce va fi creată. Se poate alege dintre: **forward lookup zone, forward and reverse lookup zones** sau **root hints**. Se va crea o zonă de rezolvare directă (*forward lookup*).



7-4: Alegerea zonei

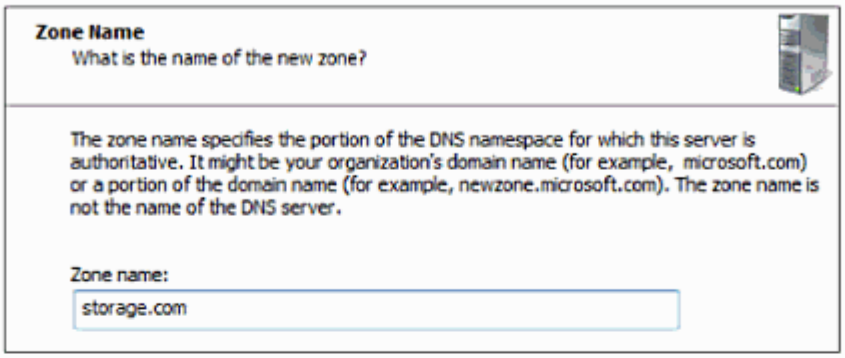
2. În continuare, se specifică dacă serverul local este responsabil pentru zonă sau deţine doar o copie *read-only* a zonei configurate pe un alt server. Se merge pe prima opţiune.



Material cu drept de autor

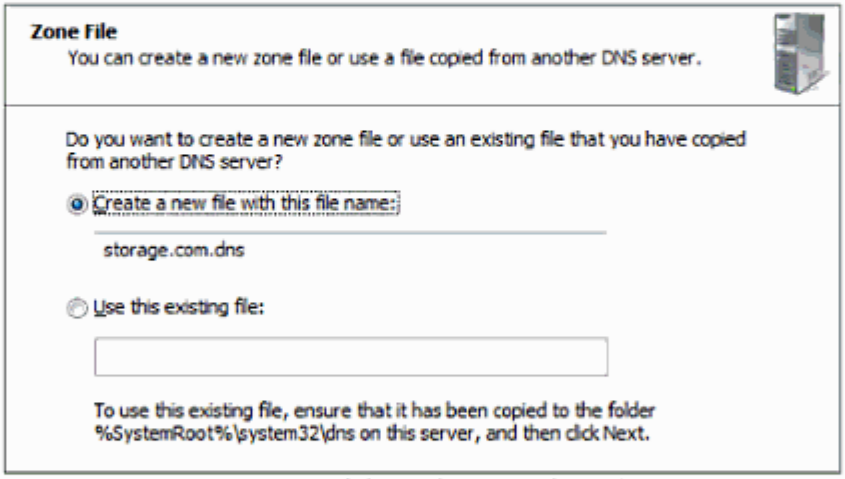
7-5: Alegerea serverului principal pentru zonă

3. Se introduce un nume pentru zona creată. Acesta este FQDN-ul său (*Fully Qualified Domain Name*).



7-6: Numele zonei

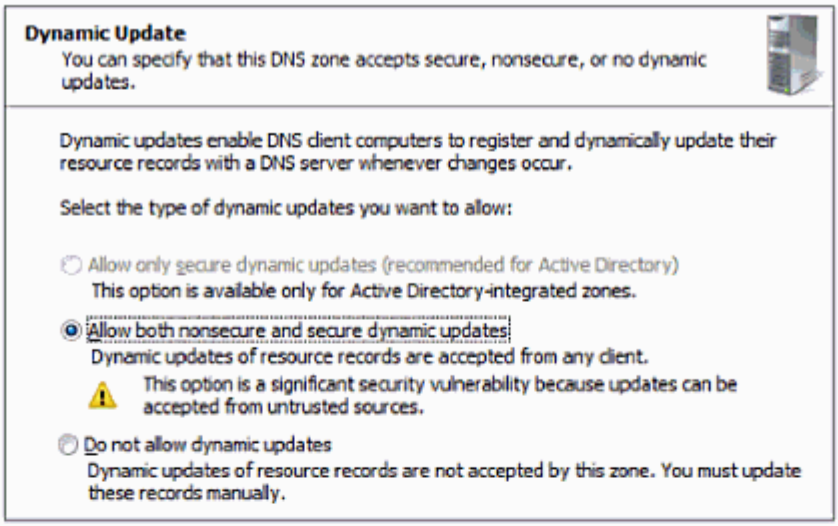
4. Fişierul de configuraţie al zonei poate fi unul nou sau există opţiunea de a încărca o configuraţie dintr-un fişier deja existent, creat pe un alt server. Fişierele de configuraţie sunt de tip text, în format ASCII şi sunt localizate în `%systemroot%\system32\dns`. Se va crea un fişier nou:



7-7: Fişierul de configuraţie al zonei

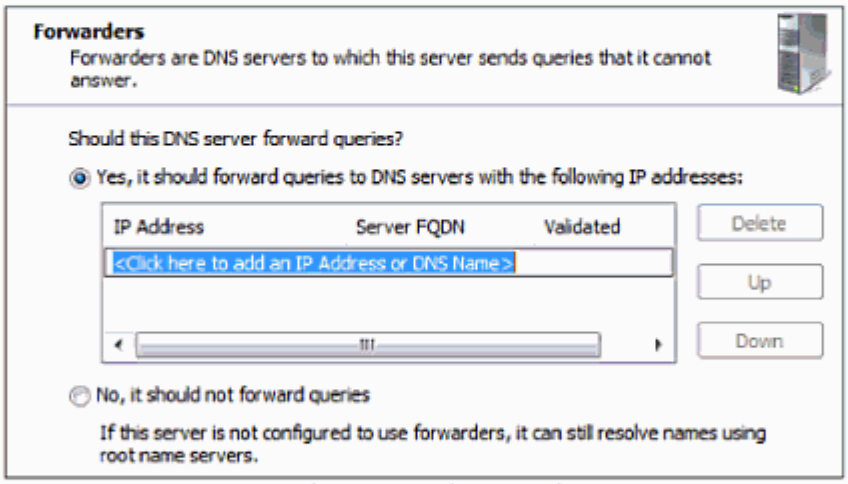
5. Următoarea pagină specifică dacă zona va accepta sau nu actualizări dinamice. Dacă actualizările dinamice sunt active, clienţii DNS vor putea să îşi înregistreze şi să îşi actualizeze propriile întrări (RR – *Resource Records*). În acest caz este important de verificat identitatea surselor de la care sosesc aceste actualizări pentru a evita inserarea de informaţii corupte în fişierele de configuraţie. Pentru acest exemplu se va alege *Allow both secure and nonsecure updates*:

Material cu drept de autor



7-8: Actualizări dinamice

6. La pasul următor se alege comportamentul pentru cererile pe care serverul nu le poate rezolva direct. Există opțiunea de a *forward*-a cererile spre alte servere sau de a încerca să rezolve astfel de cereri începând de la *root servers*, fără a le *forward*-a mai departe. Se alege opțiunea cea din urmă:



7-9: Forward-area cererilor spre alte servere

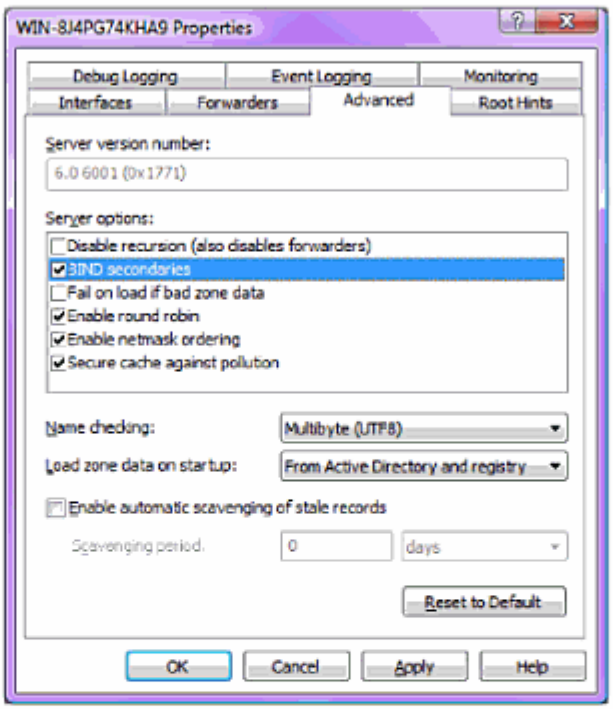
7.4.1.2 Proprietăți ale serverului DNS

Pentru a accesa proprietățile avansate ale serverului DNS se alege opțiunea *Properties* din meniul contextual al serverului (obținut la clic dreapta pe numele serverului, din interfața de administrare).

În fereastra de proprietăți a serverului sunt disponibile opțiunile grupate în următoarele categorii:

- **Interfaces**: Cuprinde interfețele de rețea pe care serverul DNS va asculta cereri. Pot fi selectate automat toate interfețele active sau se pot selecta doar anumite interfețe, după adresele IP (recomandabil, statice), IPv4 sau IPv6.

- **Forwarders**: Reprezintă aceeaşi listă de servere ce putea fi completată şi la configurarea iniţială: serverele din această listă vor fi contactate pentru cererile pe care serverul DNS nu le poate rezolva singur.
- **Advanced**: Opţiunile avansate includ activarea sau dezactivarea cererilor recursive (şi totodată a utilizării serverelor de *forwarding*), posibilitatea de compatibilitate cu serverele BIND (spre exemplu pentru cazul în care se foloseşte un astfel de server ca server secundar), comportamentul la erorile din fişierele de configurare a zonelor, *round-robin* în funcţie de priorităţi[1], setul de caractere al numelor acceptate în cereri, şamd.

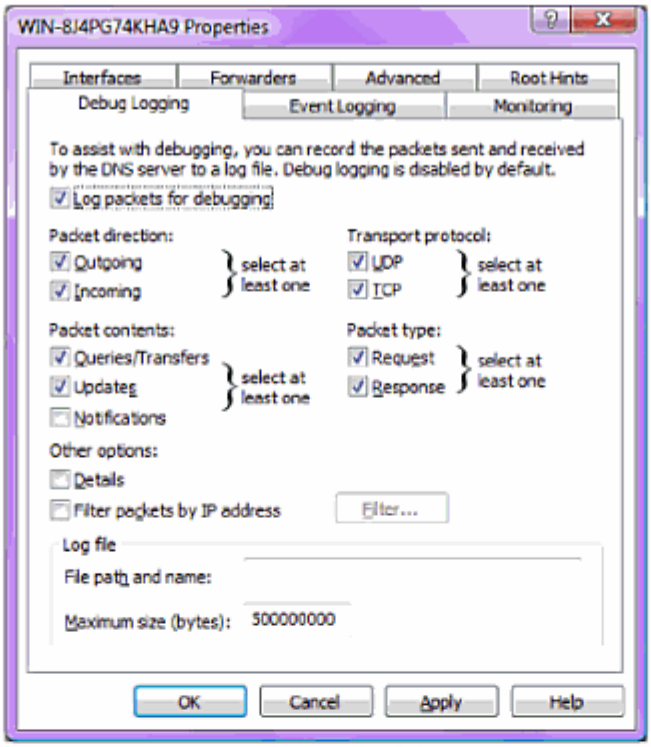


7-10: Opţiuni avansate ale serverului DNS

- **Root hints**: Lista de servere ce sunt contactate pentru zonele ce nu sunt configurate local în serverul DNS, în cazul în care nu există servere de *forwarding* sau acestea nu răspund.
- **Debug logging**: Pentru a supraveghea funcţionarea serverului, pot fi înregistrate într-un fişier informaţii despre pachetele care circulă prin serverul DNS. Se poate selecta direcţia (intrare, ieşire), protocolul folosit (TCP, UDP) şi ce tipuri de pachete vor fi notate. De asemenea, se poate aplica un filtru pentru a selecta pachetele în funcţie de adresa IP de la care sosesc sau spre care sunt destinate (Figura 7-11).
- **Event logging**: Serverul DNS poate menţiona în jurnale diferitele evenimente ce apar în decursul rulării sale, ca erori şi avertismente. Aici se poate configura care dintre acestea vor apărea în jurnale.
- **Monitoring**: Aici pot fi efectuate o serie de teste asupra funcţionării serverului, folosind cereri recursive sau nerecursive. Se pot efectua teste imediate sau la intervale regulate, iar rezultatele lor sunt evaluate.

[1] Round-robin activat din proprietăţile serverului se face la nivel de server şi nu la nivel de zonă.

Material cu drept de autor



7-11: Opţiuni de *debug*

7.4.1.3 Proprietăţi ale intrărilor DNS în *Windows Server 2008*

O zona DNS conţine diverse tipuri de intrări, numite *resource records* (RR). Ele sunt esenţa unei zone DNS, oferind informaţii despre nume, adrese şi, în unele cazuri, despre unele servicii.

Pentru adăugarea unui nou *resource record*, având selectată zona dorită, din panoul de acţiuni se alege tipul de intrare ce se doreşte a fi creată.

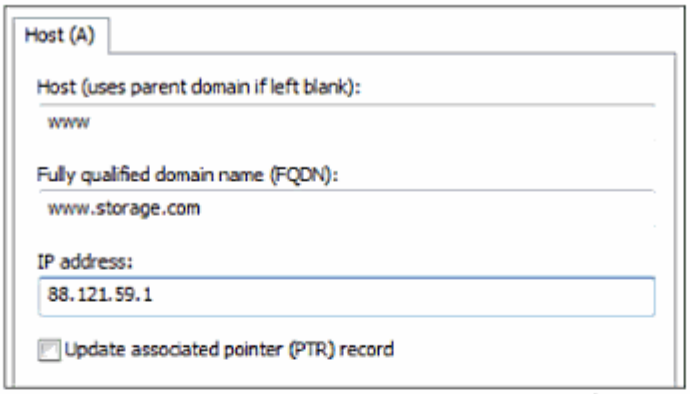
Tipurile de intrări suportate de serverul DNS pe *Windows Server 2008* se conformează standardului descris în *RFC 1035 – Domain names: Implementation and specification*:

- **A (host)**: Mapează o un nume la o adresă IP. O intrare de tipul A în fişierul de configuraţie arată în felul următor:

```
storage A 192.168.1.12
```

Folosind intrările A, se poate implementa o tehnică de *load balancing* denumită şi *round-robin DNS*, în care, sunt definite mai multe intrări de tip A cu acelaşi nume dar adrese IP diferite, astfel că un client care interoghează serverul DNS are o anumită probabilitate de a primi adresa unui anumite staţii. Tehnica este utilă în momentul în care se doreşte scăderea încărcării asupra unui server prin instalarea mai multor servere ce oferă acelaşi serviciu şi reprezintă o soluţie foarte simplă (deşi uneori ineficientă, în practică) de distribuire uniformă a cererilor spre servere. Tot în practică, trebuie ţinut cont de faptul că un client *Windows 2000* sau XP va păstra în *cache*-ul local informaţia despre primul server a cărui adresă a obţinut-o în urma rezolvării. O astfel de intrare are valoarea implicită de 86400 secunde (o zi). Micşorarea acestei valori poate pune mai bine în valoare *load-balancing*-ul implementat pe serverul DNS. Totuşi, acest gen de *load balancing* are dezavantajele sale: nu e flexibil, nu ţine cont de capabilităţile serverelor între care distribuie cererile şi nu poate fi notificat despre un server care devine inoperaţional.

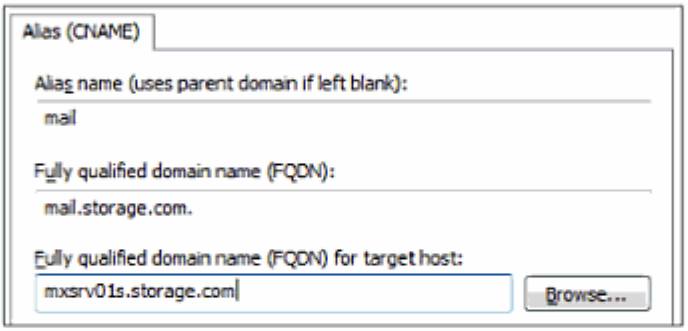
Material cu drept de autor



7-12: Proprietăţile unei intrări de tip A (host)

- **CNAME (canonical name)**: Permite atribuirea mai multor nume aceleiaşi staţii, ce foloseşte o singură adresă IP. Prin CNAME, o staţie poate fi accesată printr-o adresă IP şi mai multe alte nume. Un CNAME reprezintă, practic, un alias. O intrare CNAME în fişierul de configuraţie arată în felul următor:

```
ftp CNAME data.storage.com
```



7-13: Proprietăţile unei intrări de tip CNAME

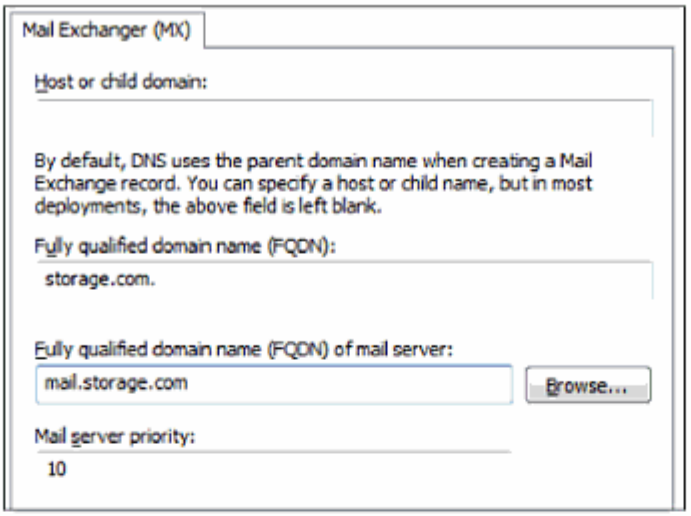
- **MX (mail exchanger)**: Înregistrează identitatea serverului (sau serverelor) de e-mail pentru o anumită zonă sau domeniu. Ele direcţionează toate calculatoarele ce doresc să trimită mesaje e-mail spre un anumit domeniu spre un anumit server destinat primirii lor. În fişierele de configurare, intrările MX sunt identificate printr-un număr de preferinţă. Numerele mai mici oferă o prioritate mai mare. Exemple de intrări MX sunt următoarele:

```
@ MX 10 mail.storage.com
@ MX 100 queue.storage.com
```

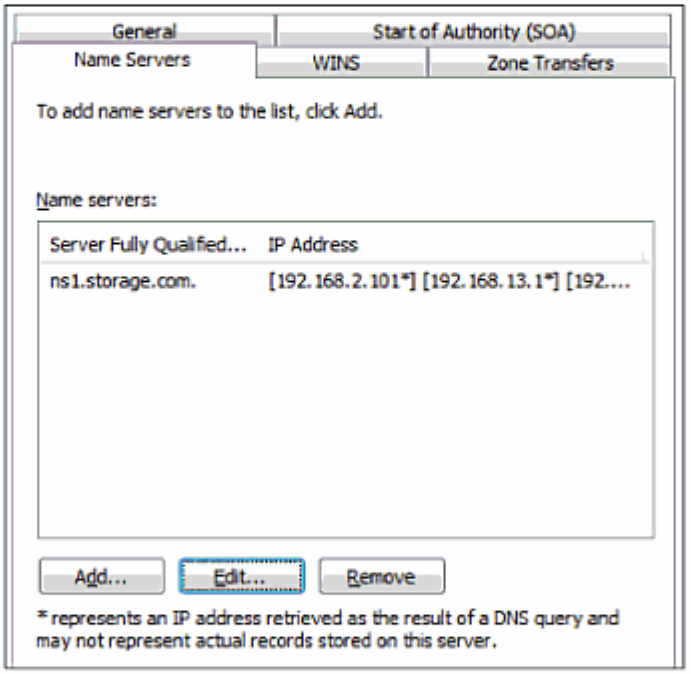
  În cazul în care primul server nu răspunde, va fi contactat cel de-al doilea, pentru că numărul său de preferinţă este mai mare. Există posibilitatea de a defini mai multe intrări de tip MX cu aceeaşi valoare numerică drept preferinţă, ceea ce are drept efect realizarea unui *load balancing* simplu, ca şi în cazul intrării A.
- **NS (nameserver)**: Intrările definesc serverele de nume ce răspund cererilor pentru un anumit domeniu. De asemenea, ele pot delega rezolvările numelor pentru anumite subdomenii unor altor servere. O intrare NS într-un fişier de configurare arată în modul următor:

```
@ NS ns1.storage.com
```

Material cu drept de autor



7-14: Proprietăţile unei intrări de tip MX

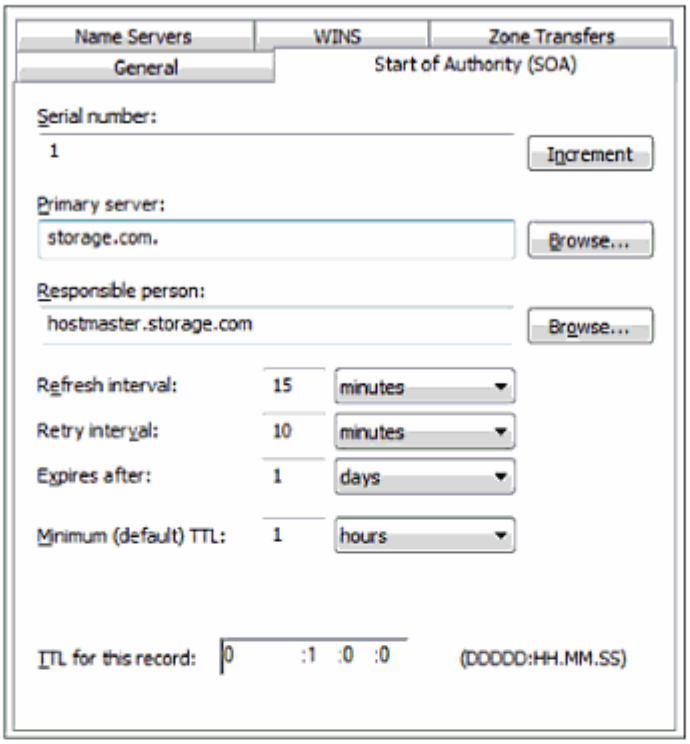


7-15: Proprietăţile unei intrări de tip NS

- **SOA (start of authority)**: Cuprinde serverele de nume primare ce sunt autoritare pentru o anumită zonă, precum şi o informaţie de contact pentru administratorul respectivei zone. De asemenea, stabileşte perioada de timp pentru care un server neautoritar poate păstra informaţiile primite înainte de a le verifica din nou la serverul autoritar. Un exemplu de intrare SOA este umrătorul:

```
@ IN SOA ns.storage.com. admin.storage.com. (
200808272000; serial number - timestamp al ultimei modificări
100; refresh
50; retry
86400 ; expire
3600 ) ; default TTL - validitatea informaţiilor de la serverul autoritar
```

Material cu drept de autor



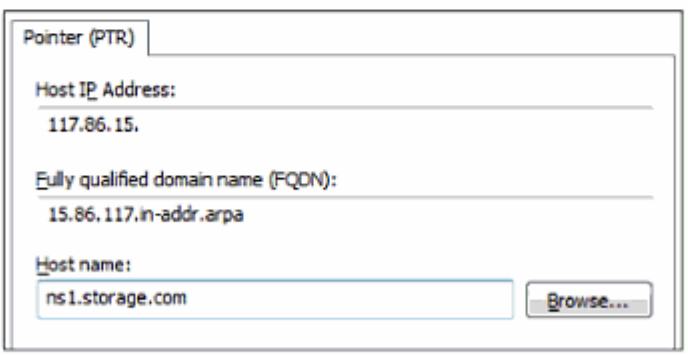
7-16: Proprietățile unei intrări de tip SOA

O intrare SOA este automat creată la instalarea serverului DNS, cu valori implicite pentru mașina locală.

- **PTR (pointer)**: Realizează funcția opusă unei intrări de tip A. Un exemplu poate fi următorul:

```
61.130.98.66.in-addr.arpa. IN PTR site3.storage.com
```

Pentru a defini o intrare de tip PTR (manual sau automat, la crearea unei intrări de tip A) este necesară crearea unei zone de tip *reverse lookup* (zonă de rezolvare inversă). Crearea unei astfel de zone se face similar cu a uneia de tip *forward lookup*, cu diferența că va exista și opțiunea de a crea o zonă de rezolvare inversă pentru adrese IPv4 sau IPv6.



7-17: Proprietățile unei intrări de tip PTR

- **SRV (service)**: Indică tipul și acoperirea unor servicii pentru o anumită zonă și sunt de mare importanță pentru funcționarea *Active Directory*. Ca și intrările de tip MX, intrările SRV dețin un număr de preferință, deci există și posibilitatea de a realiza o formă de *load balancing*. Un exemplu de intrare de tip SRV, într-un fișier de configurare, poate fi următorul:

```
_kerberos._tcp._sites.dc._msdcs 600 SRV 100 88 global.storage.com.
```

Material cu drept de autor

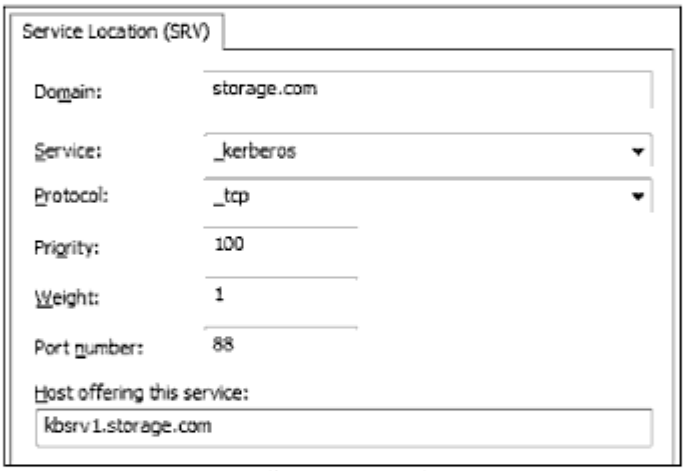
Pe scurt, câmpurile au următoarele semnificaţii:

- **_kerberos** reprezintă serviciul despre care se oferă informaţii;
- **_tcp** indică indică faptul că acesta utilizează TCP pentru funcţionare, deci poate indica şi UDP;
- **global.storage.com** reprezintă numele serverului ce oferă acest serviciu;
- **600** indică perioada de validitate (TTL) a înregistrării, în secunde;
- **88** este portul pe care funcţionează serviciul;
- **100** este un număr ce indică preferinţa acestei intrări, ca şi în cazul MX-urilor.

Intrările de tip SRV sunt importante pentru *Active Directory* deoarece indică ce staţii din domeniu rulează servicii *Active Directory*. Serviciile pe care *Active Directory* le caută în intrările de tip SRV sunt următoarele (intrarea SRV, în general, suportă mai multe):

- **_kerberos**: autentificare folosind servere *Kerberos Key Distribution Center* (KDC);
- **_kpasswd**: mecanism de schimbare securizată a parolei *Kerberos*;
- **_ldap**: *Lightweight Directory Access Protocol*, o modalitate prin care programele externe pot comunica şi schimba date cu *Active Directory*;
- **_gc**: *Global Catalog*, conţine un subset al atributelor obiectelor dintr-o infrastructură *Active Directory*.

Pentru intrările de mai sus, în cazul în care se doreşte interoperabilitatea cu servere DNS din mediul UNIX (precum BIND), nu se pot folosi caracterele – sau _.



7-18: Proprietăţile unei intrări de tip SRV

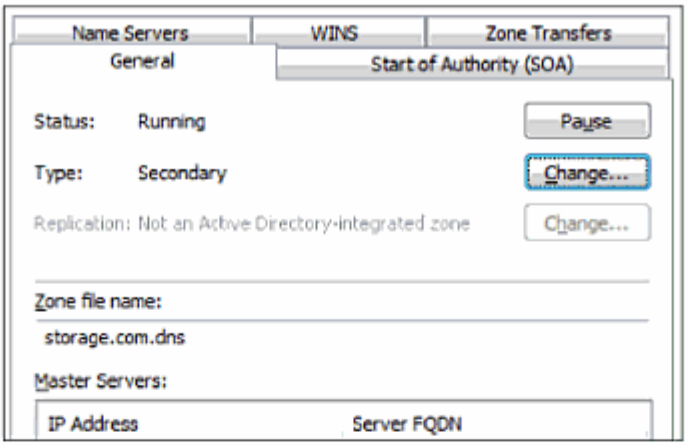
7.4.1.4 Crearea unui server de nume secundar

Pentru funcţionarea unui server de nume secundar, este necesar ca acesta să ruleze *Windows Server 2008* cu serviciul DNS instalat şi trebuie să fie configurat pentru a se folosi pe el însuşi drept server DNS:

1. Din interfaţa de administrare a serviciului DNS, sub categoria *Forward lookup zones*, se alege *New Zone* din meniul contextual;
2. Se alege *Secondary* pentru a crea o zonă de rezolvare secundară. Aceasta va indica şi faptul că serverul local este unul secundar;
3. Se introduce numele zonei secundare;
4. Se introduc pe rând numele (sau adresele) serverelor principale de pe care vor fi descărcate informaţiile zonei.

Dacă se doreşte convertirea unui server secundar în unul principal, se poate realiza aceasta selectând zona definită ca secundară şi accesând opţiunea *Properties* din meniul contextual. La categoria *General*, în dreptul lui *Type*, se apasă butonul *Change* (figura 7-19), după care se poate alege noul tip de zonă: *Primary*, *Secondary* sau *Stub*.

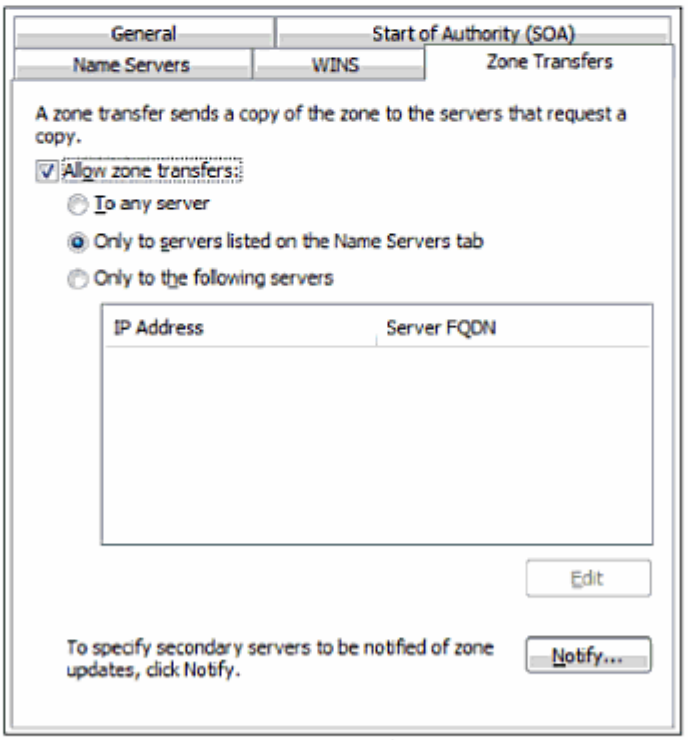
Material cu drept de autor



7-19: Convertirea unei zone secundare

7.4.1.5 Transferuri de zone

Din moment ce utilizatorii obişnuiţi din Internet nu trebuie să aibă dreptul de a obţine copii ale zonelor de pe orice server DNS deoarece aceasta ar putea expune întreaga infrastructură a serverelor unei reţele, deci un risc mare de securitate, se poate controla modul în care se pot realiza transferurile de zone. Interfaţa pentru aceasta se accesează prin meniul de proprietăţi al unei zone, sub *Zone transfers* (figura 7-20):



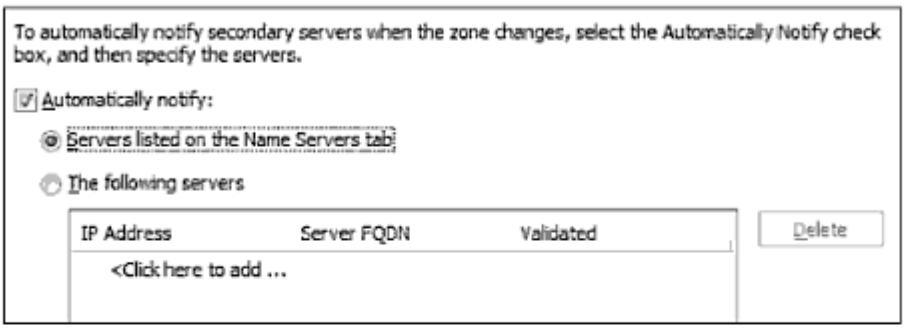
7-20: Configurarea transferurilor de zone

După cum se observă, transferurile de zone pot fi dezactivate în totalitate. Totuşi, dacă se alege activarea lor, se pot impune limitări astfel încât acestea să nu poată fi realizate decât cu serverele listate la *Name servers* (deci serverele de nume aflate sub control propriu, cel mai

Material cu drept de autor

adesea) sau doar cu o serie de servere ale căror adrese pot fi introduse în lista de mai sus (fig 7-20).

Tot de aici poate fi configurată și opțiune de notificare a altor servere în urma schimbărilor efectuate într-o zonă. Pot fi contactate doar serverele din lista *Name Servers* sau efectiv cele introduse manual în lista din fereastra *Notify* (figura 7-21):



7-21: Notificarea altor servere

7.5 Configurări în linie de comandă

7.5.1 Fișierul Hosts

Ca și în cazul sistemelor Linux, și Windows deține un fișier prin care sunt traduse local corespondențele dintre nume și adrese IP. Acest fișier este localizat în `%systemroot%\System32\drivers\etc\Hosts` (adică, spre exemplu, în `C:\Windows\System32\drivers\etc\Hosts`).

Informațiile din fișierul `Hosts` afectează rezolvarea numelor asupra întregului sistem, nu doar pentru o conexiune, și nici doar pentru IPv4.

Înregistrările din fișierul `Hosts` sunt trecute sub forma `<adresă> <nume>` iar comentarea liniilor se face prin caracterul `#`.

7.5.2 Ipconfig

Câteva dintre funcțiile utilitarului `ipconfig`, folosit în principal pentru TCP/IP adresează și configurația DNS a sistemului local:

```
ipconfig /displaydns
```

Comanda de mai sus afișează înregistrările în *cache*-ul local DNS (practic, numele deja rezolvate și care nu au expirat încă) și este utilă pentru depistarea erorilor de rezolvare pentru anumite nume. În funcție de sistemul de operare, fiecare nume a cărui rezolvare a fost efectuată cu succes este reținut o perioadă de către sistemul de operare pentru a nu mai necesita o nouă contactare a serverului DNS pentru o eventuală nouă cerere.

```
ipconfig /flushdns
```

Folosind `ipconfig` împreună cu parametrul `/flushdns` se poate forța ștergerea intrărilor din *cache*-ul local. Toate rezolvările ulterioare vor contacta prima oară serverul DNS, construind astfel din nou *cache*-ul.

```
ipconfig /registerdns
```

Parametrul `/registerdns` forțează clientul să se reînregistreze în mod dinamic la serverul DNS, dacă zona respectivă suportă actualizări dinamice (a se vedea și secțiunea *7.4.1.1 - Configurarea inițială și crearea unei zone*).

Material cu drept de autor

7.5.3 Dnscmd

Utilitarul `dnscmd` reprezintă varianta în linie de comandă a interfeței de administrare a serverului DNS din *Windows Server 2008*. El permite administratorilor să creeze zone, să modifice înregistrări și să efectueze o serie de acțiuni administrative asupra serverului DNS. Pentru a enumera parametrii comenzii și efectele lor, se poate introduce comanda[1]:

```
dnscmd /?
```

Un exemplu de utilizare este următoarea comandă:

```
dnscmd s6.corp.marketing.local /ZoneAdd corp.marketing.local /Primary /file
corp.marketing.local.dns
```

Comanda de mai sus are ca efect crearea unei noi zone primare standard, numită `corp.marketing.local` pe un server cu numele `s6.corp.marketing.local`, a cărei configurație va fi stocată în fișierul `corp.marketing.local.dns`. În mod intuitiv, pentru crearea unei zone secundare se înlocuiește parametrul `/Primary` cu `/Secondary`.

Un exemplu de adăugare a unei noi înregistrări (*RR*) la o zonă existentă este următoarea comandă:

```
dnscmd s6.corp.marketing.local /RecordAdd corp.marketing.local www A 192.168.1.23
```

Prin comanda anterioară se adaugă o nouă înregistrare de tip A (*host*) numită `www`, la zona `corp.marketing.local`, ce face legătura cu adresa IP 192.168.1.23. Serverul pe care se efectuează modificarea este același ca și în exemplul anterior, `s6.corp.marketing.local`.

Vizualizarea tuturor zonelor dintr-un server se poate face prin comanda[2]:

```
C:\Users\Administrator>dnscmd ::1 /enumzones

Enumerated zone list:
        Zone count = 3

 Zone name                      Type       Storage         Properties

 .                              Cache      File
 15.86.117.in-addr.arpa         Primary    File            Update Rev
 storage.com                    Primary    File            Update


Command completed successfully.
```

Ștergerea *cache*-ului de pe un anumit server folosind `dnscmd` se poate face utilizând comanda:

```
dnscmd s6.marketing.local /clearcache
```

Pentru o oprire și o repornire rapidă a unui anumit server:

```
dnscmd s6.marketing.local /restart
```

Exportarea configurației unei zone DNS într-un fișier se face cu parametrul `/zoneexport`, urmat de numele zonei și apoi de numele fișierului în care aceasta va fi exportată:

```
dnscmd /zoneexport corp.marketing.local corp.marketing.local.dns
```

Ștergerea unei zone de pe un anumit server se realizează cu următoarea comandă, unde primul parametru reprezintă numele serverului pe care se efectuează modificarea, iar numele de după `/zonedelete` reprezintă numele zonei de șters:

```
dnscmd s6.corp.marketing.local /zonedelete corp.marketing.local
```

[1] O listă mai cuprinzătoare a parametrilor suportați de dnscmd poate fi accesată la adresa: http://www.minasi.com/newsletters/nws0803a.htm

[2] ::1 este notația adresei IPv6 pentru localhost, echivalentul lui 127.0.0.1 în IPv4.

Material cu drept de autor

7.5.4 Nslookup

Utilitarul `nslookup` este printre cele mai utile şi uşor de utilizat metode de testare a funcţionalităţii unui server DNS. Funcţia lui de bază este, practic, cea de rezolvare a unei cereri folosind serverul DNS declarat implicit în Windows.

Pentru o simpla interogare de tip A (*host*) se introduce comanda `nslookup` urmată de numele de rezolvat:

```
C:\Users\Administrator>nslookup cs.pub.ro
Server:  dns-cache-3.rcs-rds.ro
Address:  82.76.253.115

Non-authoritative answer:
Name:    cs.pub.ro
Address:  141.85.37.5
```

`nslookup` suportă şi alte tipuri de interogări, precum MX sau SOA. Pentru a emite astfel de interogări, se introduce comanda `nslookup` fără parametri, se specifică tipul de interogare dorit prin comanda `set query` si apoi se introduce numele de rezolvat, ca în exemplul următor (pentru a ieşi din *prompt*-ul `nslookup` se pot da comenzile `exit`, `quit` sau `CTRL-C`):

```
C:\Users\Administrator>nslookup
Default Server:  dns-cache-3.rcs-rds.ro
Address:  82.76.253.115

> set query=mx
> cs.pub.ro
Server:  dns-cache-3.rcs-rds.ro
Address:  82.76.253.115

Non-authoritative answer:
cs.pub.ro       MX preference = 5, mail exchanger = mail.cs.pub.ro
> set query=soa
> cs.pub.ro
Server:  dns-cache-3.rcs-rds.ro
Address:  82.76.253.115

Non-authoritative answer:
cs.pub.ro
        primary name server = ns.cs.pub.ro
        responsible mail addr = admin.cs.pub.ro
        serial  = 2008041301
        refresh = 28800 (8 hours)
        retry   = 7200 (2 hours)
        expire  = 604800 (7 days)
        default TTL = 86400 (1 day)
```

Material cu drept de autor

Întrebări

1. Care din afirmațiile de mai jos sunt adevărate? (alegeți toate variantele care se potrivesc)

- ❑ *DNS este o bază de date distribuită*
- ❑ *DNS este o bază de date centralizată*
- ❑ *administrarea bazei de date DNS se face centralizat*
- ❑ *administrarea bazei de date DNS se face distribuit*

2. www.kde.org poate fi numele de domeniu pentru: (alegeți toate variantele care se potrivesc)

- ❑ *O intrare de tip NS*
- ❑ *O intrare de tip A*
- ❑ *O intrare de tip PTR*
- ❑ *O intrare de tip MX*

3. Răspunsuri autoritare pot fi date de (alegeți toate variantele care se potrivesc):

- ❑ *Serverele de nume master*
- ❑ *Serverele de nume de tip forwarding*
- ❑ *Serverele de nume slave*
- ❑ *Serverele de nume tip caching-only*

4. Care sunt doi factori importanți pentru alegerea timpului de viață a unei intrări din baza de date DNS? (alegeți două variante)

- ❑ *Timpul de răspuns*
- ❑ *Lungimea numelui*
- ❑ *Numărul de separatori (.) din nume*
- ❑ *Acuratețea răspunsului*

5. Un client DNS trimite serverului DNS cereri ... ... și primește răspunsuri ... ... (alegeți toate variantele care se potrivesc)

- ❑ *nerecursive, autoritare*
- ❑ *nerecursive, neautoritare*
- ❑ *recursive, autoritare*
- ❑ *recursive, neautoritare*

6. Intrarea ce identifică serverul de nume asociat cu un domeniu este de tip:

- ❑ *A*
- ❑ *PTR*
- ❑ *MX*
- ❑ *NS*

7. Care este utilitatea seriei bazei de date a unui domeniu (serial)?

- ❑ *Este folosită de serverul master pentru a verifica consistența cu root serverele*
- ❑ *Este folosită de serverul slave pentru a ști ce s-a schimbat la serverul master*
- ❑ *Este folosită de serverele de nume pentru a identifica intrările din cache*
- ❑ *Este folosită de BIND pentru a identifica schimbările efectuate asupra fișierului principal de configurație*

Material cu drept de autor

8. Care este utilitatea câmpului TTL dintr-o înregistrare SOA?

- *Înregistrările de tip SOA nu conțin câmpuri TTL*
- *Specifică durata de viață pentru răspunsurile pozitive din cache*
- *Specifică durata de viață pentru răspunsurile negative din cache*
- *Specifică durata de viață pentru răspunsurile negative si pozitive din cache*

9. Care poate fi cauza nerezolvării cererilor DNS dacă un client are configurat un server de nume în fișierul /etc/resolv.conf, iar serverul funcționează?

- *Fișierul /etc/hosts nu conține numele interogat*
- *Fișierul /etc/nsswitch.conf nu conține cuvântul cheie DNS în linia ce descrie modalitatea de rezolvare a numelor stațiilor*
- *Nu a fost pornit serviciul de rezolvare*
- *Serviciul de rezolvare este pornit, dar portul este filtrat cu ajutorul unui firewall*

10. Care este directiva BIND9 prin care se pot specifica clienții care pot face cereri către server?

- *allow-answer*
- *allow-query*
- *forwarders*
- *forwarding-servers*

Material cu drept de autor

8 E-mail

"Diamonds are forever. E-mail comes close."

June Kronholz

Ce se învaţă din acest capitol?

- Funcţionarea serviciului de e-mail
- Protocoale folosite: SMTP, IMAP, POP3
- Configurarea Postfix
- Configurare Courier-IMAP
- Configurarea Procmail

Cine este...

Eric Paul Allman *este creatorul programului sendmail şi al precursorului acestuia numit delivermail. Sendmail a devenit o parte importantă a distribuţiei de software de la Berkley (BSD) şi este în continuare cel mai utilizat MTA în sistemele Unix. În 1998 a fontat Sendmail Inc. pentru a lucra la îmbunătăţirea sendmail.*

Wietse Venema *este un programator şi fizician olandez, creatorul şi principalul dezvoltator al server-ului de e-mail Postfix. De asemenea, a scris mai multe aplicaţii în domeniul securităţii sistemelor. Din 1996, lucreaza în Statele Unite la IBM Thomas J. Watson Research Center unde continua sa dezvolte Postfix. A primit numeroase premii pentru activităţile sale.*

Ray Tomlinson *este persoana care a implementat primul sistem de e-mail într-o reţea, mai precis în reţeaua ARPANet. Sistemul putea sa trimită e-mail-uri pentru utilizatori legaţi la computere din ARPANet folosind semnul @ pentru a separa numele utilizatorului de numele host-ului. A ajutat la implementarea protocolului Telnet şi a modificat programul SNDMSG pentru a permite trimiterea mesajelor şi la utilizatori pe altecomputere sub forma de e-mail-uri. A primit numeroase premii pentru activităţile sale.*

Electronic mail (poştă/mesagerie electronică), abreviat e-mail sau e-mail, este o metodă de compunere, transmitere, recepţie şi accesare a mesajelor peste sisteme de comunicaţie electronică. Termenul se aplică atât sistemelor de e-mail din Internet bazate pe SMTP (*Simple Mail Transfer Protocol*) cât şi sistemelor de grupuri de colaborare care permit utilizatorilor unei companii sau organizaţii să transmită mesaje alteia. În acest capitol sunt prezentate detalii legate de funcţionarea şi arhitectura sistemului de e-mail şi protocoalele şi aplicaţiile utilizate.

Ideea de mesagerie electronică datează din anul 1971, când Ray Tomlinson dezvolta prima aplicaţie de e-mail pentru ARPANET. Utilizat preponderent la începutul Internetului, serviciul de e-mail a pierdut teren în faţa altor protocoale precum FTP, HTTP, Bittorrent. Cu toate acestea, extinderea Internetului şi apariţia unor forme diverse de platforme de colaborare asigură folosirea la scară largă a serviciului. Dacă la început comunicaţia prin e-mail se realiza în cea mai mare parte între două persoane, o bună parte din mesajele trimise în Internet sunt mesaje pentru liste de discuţii, forumuri, sisteme de *ticketing* sau *bug-tracking* sau pentru alte forme de colaborare.

Probabil că cea mai importantă problemă a serviciului de e-mail sunt mesajele nesolicitate (e-mail *spam*). Se apreciază că mesajele nesolicitate reprezintă peste 80% din totalul de mesaje din lume. Diferite tehnici *anti-spam* precum integrarea lor în serverele de transfer, *DNS blacklisting*, *greylisting* sunt folosite pentru a bloca mesajele nesolicitate.

Material cu drept de autor

8.1 Arhitectură și funcționare. Protocoale

Deși serviciul de e-mail are o funcționare clasică de tipul client-server peste TCP, un mesaj parcurge mai multe componente din momentul compunerii până în momentul citirii. Componentele parcurse de mesaj din momentul livrării acestuia până la stocarea acestuia sunt denumite și agenți.

- **MTA** (*Mail Transfer Agent, server SMTP*) (denumit **mail server** sau **mail exchange server** în context DNS) - este un program sau agent software care asigură transferul mesajelor de la un calculator la altul (de la sursă la destinație);
- **MUA** (*Mail User Agent*) (denumit **e-mail client**) - este programul folosit pentru citirea, compunerea și transmiterea de mesaje de poștă electronică; citirea de mesaje se face folosind POP3 sau IMAP, iar transmiterea se face folosind SMTP (prezentate mai jos);
- **MDA** (*Mail Delivery Agent*, sau LDA - *Local Delivery Agent*) - este program care acceptă mesajele e-mail și le distribuie către căsuța poștală a destinatarului;
- **căsuță poștală** (*message storage*) – este o intrare în sistemul local de fișiere utilizată pentru stocarea mesajelor de poștă electronică; căsuța poștală este interogată de MUA pentru preluarea mesajelor de poștă electronică si de MTA (eventual MDA) pentru stocarea de noi mesaje;
- **server POP3/IMAP** – server folosit pentru copierea sau accesarea mesajelor stocate în căsuța poștală; clientul (MUA) folosește POP3 sau IMAP în cadrul comunicației cu serverul.

Serviciul de poștă electronică funcționează pe baza a trei protocoale importante, care asigură comunicația între componentele de mai sus. Toate trei protocoalele funcționează peste TCP.

- **SMTP** (*Simple Mail Transfer Protocol*) - protocol utilizat de mail servere (MTA); SMTP este standardul *de facto* pentru transmiterea mesajelor electronice în Internet; portul implicit utilizat este 25; MTA-ul ascultă pe portul specificat cereri de transmitere de mesaje de poștă electronică în format SMTP; SMTP este folosit în cadrul unei sesiuni de comunicație între MUA și MTA sau între două MTA-uri;
- **POP3** (*Post Office Protocol version 3*) - protocol utilizat de clientul de e-mail (MUA) pentru a descărca mesajele de poștă electronică de pe un server; portul implicit utilizat este 110;
- **IMAP** (*Internet Message Access Protocol*) - protocol folosit de MUA pentru accesarea mesajelor electronice pe un server; portul implicit utilizat este 143.

Pe un sistem care rulează un daemon/serviciu POP3 sau IMAP, un client de e-mail se va conecta la portul specific și, folosind protocolul în cauză, va interoga căsuța poștală a utilizatorului specificat pentru citirea mesajelor.

Formatul unei adrese de poștă electronică este format din trei campuri:

- numele utilizatorului care dorește transmiterea sau recepția mesajului;
- simbolul „@" (citit *at* sau *a-rond*);
- numele domeniului (DNS name) pentru care se transmite/recepționează mesajul.

8.1.1 Funcționarea serviciului de e-mail

Pentru a exemplifica modul în care componentele prezente mai sus interacționează, se presupune următoarea situație: Ana are adresa de mail *ana@avatar.org* și dorește să-i transmită un mesaj lui Bogdan, care are adresa de mail *bogdan@berserk.org*. MTA-ul domeniului *avatar.org* este *smtp.avatar.org*, iar cel al domeniului *berserk.org* este *smtp.berserk.org*. Pașii urmați sunt următorii:

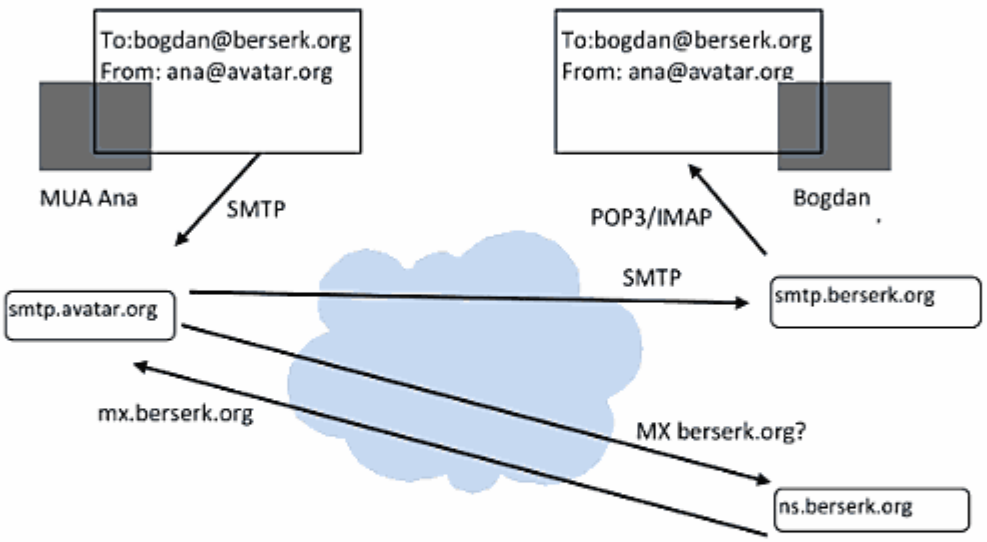
1. Ana compune un mesaj folosind un MUA; după ce comandă transmiterea mesajului, MUA-ul transformă mesajul într-un format specific[1], se conectează la MTA-ul local (sau cel configurat

[1] RFC 822

Material cu drept de autor

în MUA), în cazul de față, *smtp.avatar.org*, și folosește SMTP pentru a transmite mesajul către acesta;
2. MTA-ul analizează adresa destinație furnizată prin SMTP (*bogdan@berserk.org*); folosește DNS pentru a afla mail exchange serverul responsabil pentru domeniul *berserk.org*;
3. serverul de nume pentru *berserk.org* răspunde cu o înregistrare de tipul **MX** conținând MTA-ul responsabil: *smtp.berserk.org*; mesajul este transmis între MTA-uri (de la *smtp.avatar.org* la *smtp.berserk.org*) folosind SMTP;
4. (opțional) la destinație, LDA realizează distribuția mesajelor în căsuța poștală a lui Bogdan; LDA-ul este responsabil cu filtrarea mesajelor;
5. Bogdan folosește MUA-ul propriu pentru a citi mesajul; acesta se poate conecta pe sistemul unde se găsește căsuța de poștă electronică (folosind IMAP) sau poate ridica mesajele de pe server pentru a le consulta *offline* (folosind POP3).

Pașii de mai sus sunt prezentați în figura următoare:



8-1: Funcționarea serviciului de e-mail

8.1.1.1 Open mail relays

Inițial, MTA-urile acceptau mesaje către orice destinatar din Internet și încercau transmiterea acestora către destinație. Astfel de MTA-uri sunt denumite **open mail relays**. Acest lucru era important la începutul Internetului când conexiunile erau nesigure. Dacă un MTA nu putea ajunge la destinație, el putea transmite (*relay*) mesajul către un server *relay* care era mai apropiat de destinație. *Serverul relay* ar fi avut o șansă mult mai bună de transmitere a mesajului la un moment ulterior. Totuși, acest mecanism s-a dovedit a fi exploatabil de către cei care transmiteau mesaje electronice nesolicitate (*spam*). Drept consecință, foarte puține MTA-uri moderne au funcționalitate de *open mail relay* și cea mai mare parte nu vor accepta mesaje de la MTA-uri de tip *open mail relay* pentru că este foarte probabil ca aceste mesaje să fie *spam*.

ISP-urile practică două metode pentru a preveni ca serverele lor sa devină *open relay*: acceptarea conexiunilor doar de la clienții săi (un *spammer*, care nu este client al ISP-ului nu va putea folosi acel server) și crearea unor liste de destinatari acceptați (*allowed recipients*) astfel încât un *spammer* nu va putea folosi acel server pentru a trimite mail catre orice destinație.

Material cu drept de autor

8.1.2 Formatul mesajelor

Un mesaj electronic este format[1] dintr-un plic (*envelope*) şi conţinut. „Plicul" conţine informaţii cu privire la livrarea mesajului către destinaţie, în timp ce conţinutul mesajului este obiectul efectiv transferat. Plicul este descris, de obicei, prin comenzi SMTP:

```
MAIL FROM:<userx@y.foo.org>
RCPT TO:<@hosta.int,@jkl.org:userc@d.bar.org>
```

Conţinutul unui mesaj cuprinde două părţi: **antetul** şi **corpul** mesajului. Antetul, la rândul său, reprezintă o sintaxă strictă, în vreme ce corpul reprezintă o parte opţională în conţinutul mesajului. Corpul este format dintr-o succesiune de linii de text fără niciun fel de constrângeri.

O linie din **antet** este compusă dintr-un nume de câmp urmat de ':' şi apoi descrierea câmpului. Spre exemplu, "Subject: Mesaj de test", foloseşte câmpul "Subject", cu descrierea "Mesaj de test". Descrierile de câmp pot avea un format special, dar pot fi şi nestructurate.

Câmpurile importante din cadrul unui mesaj pot fi observate în extrasul de mesaj de mai jos:

```
Return-Path: <tavi@cs.pub.ro>
X-Original-To: razvand@cs.pub.ro
Delivered-To: razvand@cs.pub.ro
Received: from ixro-ex1.ixiacom.com (unknown [212.146.94.66])
        by mail.cs.pub.ro (Postfix) with ESMTP id 7588E156B2C;
        Wed, 25 Jun 2008 14:12:46 +0300 (EEST)
Received: from [10.205.9.116] ([10.205.9.116]) by ixro-ex1.ixiacom.com with Microsoft
SMTPSVC(6.0.3790.1830);
         Wed, 25 Jun 2008 14:14:39 +0300
From: Octavian Purdila <tavi@cs.pub.ro>
Organization: Politehnica University of Bucharest
To: Razvan Deaconescu <razvand@cs.pub.ro>
Subject: Re: doctorat tavi
Date: Wed, 25 Jun 2008 14:11:41 +0300
User-Agent: KMail/1.9.9
Cc: Razvan Rughinis <rrazvan@cs.pub.ro>
References: <200806131346.46727.tavi@cs.pub.ro>
In-Reply-To: <200806131346.46727.tavi@cs.pub.ro>
MIME-Version: 1.0
Content-Type: text/plain;
  charset="iso-8859-1"
Content-Transfer-Encoding: 7bit
Content-Disposition: inline
Message-Id: <200806251411.41675.tavi@cs.pub.ro>
X-OriginalArrivalTime: 25 Jun 2008 11:14:39.0621 (UTC) FILETIME=[A8E6BB50:01C8D684]
X-mail.cs.pub.ro-MailScanner: Found to be clean
X-mail.cs.pub.ro-MailScanner-SpamCheck: not spam, SpamAssassin (score=0.847,
        required 6, autolearn=not spam, BAYES_00 0.71,
        FORGED_RCVD_HELO 0.14)
X-mail.cs.pub.ro-MailScanner-From: tavi@cs.pub.ro
```

Câmpurile `Return-Path`, `X-Original-To`, `Delivered-To`, `Received` oferă informaţii referitoare la calea parcursă de mesaj până la destinaţie. Câmpurile care încep cu „X" sunt câmpuri speciale de extensie adăugate de utilitare specializate. În exemplul de mai sus acest utilitare sunt *MailScanner* şi *SpamAssassin*.

Alte câmpuri prezente în mesaj sunt:

1. Câmpuri referitoare la data mesajului:
   - *Date:* data transmiterii mesajului (data locală când mesajul a fost trimis spre livrare de către autor).
2. Câmpuri referitoare la autor:
   - *From:* câmpul specifică una sau mai multe adrese ale autorului/autorilor mesajului într-un format standardizat;
   - *Reply-To:* permite specificarea căsuţei poştale unde autorul sugerează trimiterea răspunsului;
   - User-agent: clientul de e-mail (MUA) folosit pentru trimiterea mesajului;
3. Câmpuri referitoare la destinatar:
   - *To:* conţine lista adreselor destinatarilor mesajelor;
   - *Cc (Carbon Copy):* conţine lista adreselor persoanelor carora le va fi livrat mesajul, fără ca acesta să le fie adresat în mod direct;

[1] RFC 2822

Material cu drept de autor

- *Bcc (Blind Carbon Copy):* conține lista adreselor persoanelor cărora le va fi livrat mesajul, fără ca aceste adrese să fie vizibile celorlalți destinatari.

4. Câmpuri de identificare a mesajului:
   - *Message-ID:* un identificator unic al mesajului;
   - *In-Reply-To:* acest câmp este prezent în mesajele de răspuns la alte mesaje și conține *Message-ID*-ul mesajului la care se răspunde;
   - *References:* câmp prezent în mesajele de răspuns, conținând identificatorii altor mesaje la care mesajul curent face referire.
5. Câmpuri referitoare la conținut:
   - *Subject:* subiectul mesajului;

8.1.2.1 MIME

Dat fiind faptul că în momentul elaborării acestor specificații nu se punea problema de a transmite altceva decât text, nu s-a definit exact forma conținutului unui mesaj. Nevoia de a transmite altceva decât text în cadrul unui mesaj (audio/video) a dus la apariția unor probleme. Soluția a fost **MIME** (*Multipurpose Internet Mail Extension*)[1]. Datorită largii răspândiri a sistemului definit de RFC 822, MIME nu putea să fie gândit decât ca o adăugire la aceste specificații, nicidecum ca o înlocuire.

Utilitarul `uuencode` poate fi folosit pentru codificarea fișierelor binare în format care poate fi transmis prin e-mail:

```
razvan@valhalla:/tmp$ uuencode Test_vm_lin.zip hello.zip
begin 644 hello.zip
M4$L#!!0````(`&:HDS6\![A=F````,D````0`!4`36%K969I;&4N8VAE8VME
M<E54"0`#<#>(169FB$55> 0`Z`/H`W-V\W%T#U:P5='-3\S)4=!-YA+2"_#P
M]XNT4BA)+2[AX@**0IF<J14%^44E"CXN\3Z>3D&.09'Q`8XA'K9Z"FIJ"GKZ
M$.5)I9DY*;H%1:E67)Q<7"`QB&Z]?"Y.%0UG9TVX1OH*"KKY8#D%E3@%71\]
[...]
```

8.1.3 SMTP (Simple Mail Transfer Protocol)

SMTP[2] este protocolul utilizat de serverele de e-mail (MTA) pentru transportul mesajelor. În cadrul unei comunicații, un MTA poate juca rol de server sau de client. Server va fi acel MTA care primește mesajul iar client MTA-ul care îl trimite. Serverul de SMTP poate fi cel final (serverul de mail exchange asociat domeniului destinație) sau poate avea rol intermediar: *relay* sau *gateway*. Un server de tip *relay* este un server care nu este cel final și va transmite mesajul către un alt server. Marea parte a serverelor de e-mail la începutul Internetului erau de acest tip pentru că legăturile slabe nu puteau garanta întotdeauna o conexiune de la sursă la destinație. Server de tip *gateway* este un server configurat ca server de *mail exchange* pentru mai multe domenii care va redirecta mesajele MTA-urilor efective asociate fiecărui domeniu.

Protocolul SMTP este un protocol bazat pe stări. Atât serverul cât și clientul împart aceeași viziune asupra stării curente. O sesiune este reprezentată de comenzile inițiate de clientul SMTP și de răspunsurile date de server, sub forma unor coduri numerice însoțite de un text explicativ.

Un protocol simplu, SMTP folosește comenzi text pentru comunicare. Astfel, pentru specificarea destinatarului și expeditorului se vor folosi comenzile RCPT TO, respectiv MAIL FROM. Alte comenzi utile sunt HELO, DATA, QUIT.

Pentru a rezolva unele dintre limitările protocolului SMTP, un nou protocol a fost definit: ESMTP (*Extended SMTP*). O dată cu acest nou protocol, a fost definită și o modalitate de a determina versiunea de SMTP suportată de un server. Astfel, un client capabil de ESMTP își va începe dialogul cu serverul utilizând comanda EHLO în loc de HELO. Dacă răspunsul primit de

[1] RFC 2045-2049
[2] RFC 2821

Material cu drept de autor

la server nu este unul de eroare, clientul va folosi în continuare ESMTP, în caz contrar revenindu-se la folosirea SMTP.

8.1.3.1 SASL şi TLS

Fiind un protocol simplu, SMTP nu are integrată o facilitate de autentificare. Oricine poate trimite mesaje prin conectarea la un server SMTP. Mai mult, îşi poate ascunde foarte uşor identitatea. Pentru a integra facilităţi de autentificare în SMTP se foloseşte SASL[1] (*Simple Authentication and Security Layer*). SASL decuplează formele de autentificare de aplicaţia în sine şi poate fi folosit cu o gamă variată de protocoale.

SASL pune la dispoziţia aplicaţiei mecanisme de autentificare (*digest*, *login*, *plain*, *one time password*, etc.) şi un cadru de stocare a informaţiei de autentificare (utilizatori, parole). Într-o conexiune SMTP autentificarea se realizează prin intermediul comenzii AUTH din extensia SMTP-AUTH[2].

SASL oferă servicii de autentificare dar nu asigură securitatea informaţiei. Numele de utilizatori sau parolele transmise în cadrul sesiunii de autentificare sunt transmise în clar. Comunicaţia criptată este asigurată cu ajutorul TLS[3] (*Transport Layer Security*). TLS permite securizarea comunicaţiei folosind mecanisme de criptare cu chei publice. În general, serverul SMTP se va autentifica cu ajutorul unui certificat digital, în timp ce clienţii se vor autentifica cu unele din metodele puse la dispoziţie de SASL.

8.1.4 POP3 (Post Office Protocol)

POP3[4] este utilizat de aplicaţiile de tip MUA pentru a ridica mesajele de poşta electronică din căsuţa poştală (*message store*).

POP3 a fost proiectat pentru a permite utilizatorilor cu conexiuni intermitente (cum sunt conexiunile *dial-up*) să extragă mesajele de pe server atunci când sunt conectaţi şi apoi să le vizualizeze şi să lucreze fără necesitatea unei conexiuni. Deşi marea parte a clienţilor de mail au opţiunea de a păstra mesajele pe server, în general MUA care folosesc POP3 se conectează, ridică mesajele, le stochează pe calculatorul utilizatorului ca mesaje noi, le şterg de pe server şi se deconectează. Dezavantajul este dificultatea citirii aceloraşi mesaje din două locaţii diferite. În contrast, protocolul **IMAP** de ridicare a mesajelor electronice suportă atât mod de operare *conectat* cât şi *deconectat*.

Dialogul începe prin stabilirea conexiunii la cererea clientului. Serverul va răspunde cu un mesaj de întâmpinare. Urmează apoi un dialog format din comenzi ale clientului şi răspunsuri ale serverului. Răspunsul serverului include un indicator al stării curente (+OK sau -ERR), un cuvânt cheie şi, eventual, informaţie adiţională. La fel ca şi SMTP, comenzile POP3 sunt comenzi text.

După iniţierea conexiunii, se va realiza autentificarea clientului la server. Ca şi alte protocoale mai vechi din Internet, POP3 folosea la început mecanisme de autentificare fără criptare. Deşi încă se practică transmiterea de parole în clar, POP3 suportă câteva metode de autentificare pentru a furniza diverse niveluri de protecţie împotriva accesului nedorit la mesajele utilizatorului. O astfel de metodă este comanda **APOP** care foloseşte MD5 pentru obţinerea unei chei de autentificare. POP3 suportă de asemenea metode de autentificare de tip IMAP folosind extensia AUTH. În zilele noastre este însă comun să se cripteze traficul POP3 folosind TSL/SSL.

---

[1] RFC 4422
[2] RFC 4954
[3] RFC 4346
[4] RFC 1939

Material cu drept de autor

8.1.5 IMAP (Internet Message Access Protocol)

Protocolul POP3 este utilizat în special pentru transferarea mesajelor de pe server pe orice alt calculator pentru a le citi *offline*. În cazul în care se doreşte accesarea unui singur cont de poştă electronică de la mai multe locaţii (acasă/birou), protocolul POP3 este limitat. Acest dezavantaj a dus la dezvoltarea unui alt protocol, IMAP (*Internet Message Access Protocol*)[1]. În mod asemănător cu POP3, IMAP permite transferarea mesajelor către un alt calculator. Atunci când IMAP este utilizat în modul *online* clientul nu transferă mesajele şi nici nu le şterge de pe server. Clientul poate cere însă să primească antetele mesajelor, anumite părţi din mesaje, sau chiar mesajele care se potrivesc unui criteriu de căutare. În esenţă, IMAP permite manevrarea mesajelor dintr-o căsuţă poştală aflată pe un server ca şi când acestea ar fi stocate local.

Spre deosebire de POP3, protocolul IMAP oferă mai multe posibilităţi de prelucrare a mesajelor din căsuţa poştală. Cea mai importantă caracteristică este accesarea şi manevrarea mesajelor direct pe server. Fiind menţinute pe server, mesajele nu sunt descărcate decât la vizualizarea lor, existând şi opţiunea descărcării doar a unei părţi din mesaj. Spre deosebire de POP3, marcarea mesajelor este mult mai flexibilă. Există marcaje de: *Seen* (vizualizat), *Answered* (răspuns), *Flagged* (marcat ca fiind urgent), *Deleted* (şters), *Draft* (nu a fost terminată compunerea mesajului), *Recent* (este folosit pentru marcarea mesajelor noi, în prima sesiune în care sunt vizualizate).

Spre deosebire de POP3, unde toate mesajele se găsesc pe server într-o singură căsuţă poştală (*inbox*), iar directoarele (*mailbox*) nu se pot crea decât în aplicaţia client (MUA), protocolul IMAP permite crearea de directoare direct pe server. Utilizând comenzi IMAP, aplicaţia client poate folosi filtre pentru mutarea mesajelor dintr-un director în altul, fără să fie nevoie măcar ca *mailbox*-urile să fie situate pe acelaşi server.

O altă facilitate este posibilitatea accesării concurente a căsuţelor poştale, aceasta făcând posibilă eventual şi partajarea unei căsuţe între mai mulţi utilizatori.

Cu toate că facilităţile oferite de IMAP sunt superioare POP3, avantajul protocolului POP3 rămâne acela al simplităţii şi al nesolicitării extensive a resurselor serverului. Din aceste motive, majoritatea furnizorilor de servicii Internet (ISP) sau e-mail vor pune la dispoziţia utilizatorilor servicii POP3. De partea cealaltă, reţelele locale folosesc de multe ori IMAP. Folosind IMAP peste o reţea de viteză ridicată, mesajele sunt accesate imediat, spre deosebire de recuperarea acestora folosind POP3 sau accesarea prin intermediul unei interfeţe web.

8.2 Formatul căsuţei poştale

După primirea unui mesaj destinat staţiei locale, MTA va stoca mesajul într-o intrare din sistemul local de fişiere denumită căsuţă poştală, *message store* sau *mail spool*. Formatul căsuţei poştale trebuie să fie cunoscut de clienţii de e-mail locali care vor citi mesajele sau de serverele POP3 sau IMAP care vor accesa căsuţa pentru a livra mesajele utilizatorilor.

Deşi anumite servere folosesc formate particulare pentru căsuţa poştală, cele mai întâlnite două formate sunt `mbox` (formatul tradiţional) şi `Maildir` (format mai recent).

8.2.1 mbox

Formatul `mbox` este forma tradiţională de stocare a mesajelor pe un sistem Unix într-un singur fişier asociat fiecărui utilizator. Fiecare nou mesaj din fişier începe cu o linie care începe cu `From` urmat de un caracter spaţiu.

[1] RFC 2060

Material cu drept de autor

Formatul `mbox` este formatul folosit implicit de majoritatea MTA-urilor. În mod obişnuit, mesajele sunt livrate în directorul `/var/spool/mail/$USERNAME`[1]. MTA-urile pot fi, însă, configurate să livreze mesajele într-un fişier de tipul `mbox` din directorul `home` al utilizatorului.

Întrucât fişierul `mbox` poate fi accesat simultan atât de MTA cât şi de serverul IMAP/POP3, este necesară o formă de *locking* care să asigure consistenţa accesului. În mod evident, *locking*-ul atrage după sine penalizări de performanţă şi dificultatea implementării pe un sistem de fişiere montat în reţea (cum ar fi NFS). Aceste probleme au fost cele care au condus la crearea formatului `Maildir`.

8.2.2 Maildir

Formatul `Maildir` se deosebeşte de formatul `mbox` prin faptul că nu necesită mecanisme de *locking* pentru a asigura integritatea mesajelor în timp ce mesajele sunt adăugate, şterse, mutate. Fiecare mesaj este menţinut într-un fişier separat. Modificările sunt realizate prin intermediul operaţiilor atomice peste sistemul de fişiere.

Formatul `Maildir` a fost creat de Daniel J. Bernstein în momentul scrierii serverului `qmail`.

Directorul `Maildir` conţine trei subdirectoare: `tmp`, `new` şi `cur`, localizate, de obicei, în cadrul unui director din `home`-ul utilizatorului:

```
razvan@anaconda:~$ ls -F /home/razvan/Maildir/
courierimaphieracl/   courierimapsubscribed  courierpop3dsizelist  new/
courierimapkeywords/  courierimapuiddb       cur/                  tmp/
```

Fişierele din directorul `new` sunt mesaje livrate dar care nu au fost citite. Linia care începe cu `From` nu mai este necesară. După ce un mesaj este vizualizat este mutat în directorul `cur`. Directorul `tmp` este folosit în momentul livrării mesajelor pentru a stoca un mesaj până la scrierea acestuia în directorul `new`.

8.3 Clienţi de e-mail

Un client de e-mail sau MUA este aplicaţia folosită pentru a transmite şi a citi mesaje de poştă electronică. Transmiterea presupune contactarea unui server SMTP iar citirea înseamnă folosirea unui server IMAP sau POP3.

În zilele noastre clienţii de e-mail ocupă un spectru larg de funcţionalităţi, de la clienţi în linie de comandă (`alpine`, `mutt`) până la aplicaţii integrate de tip PIM (*Personal Information Manager*) (*Microsoft Outlook*, *Novell Evolution*) sau aplicaţii web care îndeplinesc rolul de client de e-mail (Gmail, Yahoo! Mail).

Facilităţile de bază ale unui client de e-mail sunt folosirea POP3/IMAP pentru descărcarea/accesarea mesajelor, folosirea de filtre, folosirea de semnături, autentificarea şi criptarea comunicaţiei. Printre facilităţile furnizate de clienţii de e-mail moderni se numără:

- vizualizarea firelor de discuţii
- suport PGP, S/MIME
- etichetarea mesajelor
- corectarea erorilor de gramatică
- vizualizarea imaginilor ataşate (*image preview*), căutare indexată

Exemple de clienţi de e-mail sunt:

- clienţi în linie de comandă: `mailx`, `mutt`, `alpine`
- clienţi cu interfaţă grafică: *Microsoft Outlook*, *Mozilla Thunderbird*, *Novell Evolution*, *KMail*, *Opera Mail*
- clienţi cu interfaţa web (*webmail*): Gmail, Yahoo! Mail, Horde IMP, SquirrelMail

[1] http://en.wikipedia.org/wiki/Filesystem_Hierarchy_Standard#Directory_structure

Material cu drept de autor

8.3.1 mailx

mailx (o versiune îmbunătățită a *mail*) este un utilitar pentru transmiterea şi primirea de mesaje pe sisteme Unix. Este un client de e-mail de bază, lipsit de posibilitatea de citire a mesajelor de pe alt sistem. Mesajele sunt citite de pe sistemul local.

Deşi cu puține funcționalități, *mailx* poate fi folosit pentru manverarea rapidă a mesajelor stocate pe sistemul local. Cea mai frecventă utilizare a *mailx* este transmiterea de mesaje direct din linia de comandă. Acest lucru permite folosirea *mailx* în majoritatea scripturilor care automatizează trimiterea de mesaje de poştă electronică.

Citirea mesajelor se face interactiv prin invocarea comenzii *mailx* (sau forma compatibilă *mail*):

```
alina@anaconda:~$ mail
Mail version 8.1.2 01/15/2001.  Type ? for help.
"/var/mail/alina": 24 messages 24 new
>N  1 liviu.dumitrascu@  Sat Jun  9 01:24  178/20754 61 joburi noi pentru tine
 N  2 newsletter@9am.ro  Sat Jun  9 02:47  960/53849 Casatorii pentru comunitate
 N  3 liviu.dumitrascu@  Sat Jun  9 07:46   96/7653  Horoscopul Carierei pentru
 N  4 liviu.dumitrascu@  Sun Jun 10 01:06  165/18540 17 joburi noi pentru tine
 N  5 newsletter@wall-s  Sun Jun 10 02:11  312/41081 Cele mai citite articole di
 N  6 newsletter@9am.ro  Sun Jun 10 09:25  660/52453 Cele mai citite articole di
 N  7 newsletter@wall-s  Mon Jun 11 03:20  446/38829 Luni, 11 Iunie - O saptamana
```

Orice mesaj are un index şi o stare: nou, necitit, propus pentru a fi şters, mesaj la care s-a răspuns, etc. Promptul oferit de mail este &. Comenzile posibile pot fi vizualitate prin activarea ecranului de ajutor (comanda h): t – afişează, d – şterge, r – replică.

Invocarea non-interactivă a comenzii permite transmiterea de mesaje. Astfel, dacă se doreşte transmiterea unui mesaj către utilizatorul `ana@avatar.com` cu subiectul „Filmul saptamanii", se va folosi comanda:

```
razvan@anaconda:~$ mail -s "Filmul saptamanii" ana@avatar.com
Vii la film miercuri?
.
Cc:
```

Caracterul . (sau opțional CTRL-D) înseamnă încheierea mesajului. `mailx` solicită introducerea unui destinatar de tipul *Carbon Copy*, care a fost ignorată.

Dacă mesajul este stocat într-un fişier, se poate folosi următoarea comandă pentru transmiterea sa:

```
razvan@anaconda:~$ cat hello.txt | mail -s "Filmul saptamanii" ana@avatar.com
```

Avantajul ultimei comenzi este lipsa de interactivitate. O astfel de comandă poate fi uşor integrată în scripturi care necesită transmiterea automată a mesajelor de poştă electronică.

8.4 MTA

Se apreciază că, în ziua de astăzi, peste 80% din MTA-urile existente în Internet rulează *Sendmail*, *Microsoft Exchange Server*, *Postfix* şi *Exim*.

Sendmail a fost pentru multă vreme serverul de e-mail implicit datorită dezvoltării sale încă de la începutul Internetului. *Sendmail* a fost scris de Eric Allman la începutul anilor '80. Versiunea actuală este 8.14. *Sendmail* nu a fost proiectat cu aspecte de securitate drept pentru care a fost cauza a numeroase atacuri pe parcursul dezvoltării Internetului.

Problemele de securitate ale *Sendmail* au condus la crearea *Qmail* şi *Postfix*. *Qmail*, scris de Daniel J. Bernstein, s-a dorit a fi revoluționar prin proiectarea ce ținea cont de securitate. Din păcate, începând cu 1997, *Qmail* nu mai este dezvoltat şi numărul de sisteme ce rulează *Qmail* a scăzut.

Postfix a urmărit, de asemenea, atent condițiile necesare pentru asigurarea securității. Una dintre deciziile importante a fost abandonarea sistemului monolitic folosit de *Sendmail* şi

Material cu drept de autor

folosirea unui sistem modular. *Postfix* este compus dintr-un set de daemoni cu drepturi limitate care îndeplinesc diverse sarcini necesare.

Exim folosește modelul monolitic al *Sendmail* fără a avea parte, însă, de aceeași istorie de vulnerabilități. Cunoscându-se problemele de securitate ale *Sendmail*, *Exim* a fost proiectate pentru a nu suferi de aceleași vulnerabilități. Ajuns la versiunea 4, *Exim* este MTA cu un nivel ridicat de configurabilitate. Printre funcționalitățile avansate ale *Exim* se numără folosirea de liste de acces și integrarea unui *framework* de scanare a conținutului util ca filtru antivirus sau *anti-spam*.

În lumea Unix, *Sendmail* rămâne cel mai folosit MTA. Totuși, dificultatea în configurare, istoria de vulnerabilități și existența unor soluții precum *Postfix* și *Exim* a dus la diminuarea instalărilor de *Sendmail*. În ziua de astăzi, majoritatea administratorilor de sistem recomandă folosirea *Postfix* sau *Exim*.

8.5 Postfix

Apărut inițial în cadrul unui proiect de jumătate de an pornit de Wietse Venema, *Postfix* a fost dezvoltat ulterior, dovedindu-se o alternativă mai rapidă și mai sigură pentru *Sendmail*.

Obiectivele de proiectare ale *Postfix* au fost fiabilitatea, performanța, ușurința în utilizare și administrare și securitatea. *Postfix* pune la dispoziția administratorului un număr limitat de fișiere de configurare cu directive simple. Pentru a facilita adoptarea *Postfix* ca MTA, multe dintre fișierele și comenzile folosite de *Sendmail* sunt compatibile în *Postfix*. Comenzi precum `sendmail`, `newaliases` și fișiere precum `/etc/aliases` sau `.forward` și-au păstrat rolul și în *Postfix*.

Problemele majore ale *Sendmail* au fost cele de securitate. *Postfix* folosește o serie de mecanisme pentru asigurarea securității. Una din deciziile de proiectare importante a fost modularitatea. În vreme ce *Sendmail* este un sistem monolitic, folosind un singur executabil cu drepturi privilegiate pentru executarea sarcinilor, *Postfix* folosește un proces cu drepturi limitate pentru fiecare tip de sarcină. Fiecare din procesele *Postfix*, denumite și agenți, rulează sub paradigma celui mai mic privilegiu și execută doar sarcina proprie: transmitere mesaj, stocare mesaj, livrare locală, gestiunea cozii, etc. Singurul proces care rulează cu drepturi privilegiate este procesul master care le pornește pe toate celelalte. *Postfix* folosește de asemenea facilitatea de `chroot` a sistemelor Unix pentru a limita vizibilitatea sistemului de fișiere pentru un proces. De obicei, procesele *Postfix* sunt rulate în *jail*-ul `/var/spool/postfix`.

8.5.1 Arhitectură

8.5.2 Instalare

Pe sistemele *Debian-based*, *Postfix* este disponibil în forma binară în pachetul `postfix`.

```
# apt-get install postfix
```

În cadrul procesului de instalare trebuie răspuns la câteva întrebări pentru a se specifica:

1. *Tipul serverului*. Din opțiunile disponibile (*No configuration, Internet Site, Internet with smathost, Satellite system, Local Only*), cel mai adesea se va alege *Internet Site*.
2. *Numele serverului*. Trebuie specificat sub forma *nume.domeniu*, de exemplu *mail-test.cs.pub.ro*.

După instalare se specifică modul în care se poate altera configurația curentă a *Postfix* și vizualizarea valorilor configurate:

```
        Postfix is now set up with a default configuration.  If you need to make changes, edit
        /etc/postfix/main.cf (and others) as needed.  To view Postfix configuration values, see
postconf(1).
```

Material cu drept de autor

```
After modifying main.cf, be sure to run '/etc/init.d/postfix reload'.
```

8.5.3 Interacțiunea cu Postfix

Interacțiunea cu serverul *Postfix* se poate realiza în mai multe moduri. Ca orice serviciu, *Postfix* poate fi oprit, repornit, pornit prin intermediul scripturilor de inițializare a sistemului:

```
cuirass:~# /etc/init.d/postfix stop
Stopping Postfix Mail Transport Agent: postfix.
cuirass:~# /etc/init.d/postfix start
Starting Postfix Mail Transport Agent: postfix.
cuirass:~# /etc/init.d/postfix restart
Stopping Postfix Mail Transport Agent: postfix.
Starting Postfix Mail Transport Agent: postfix.
În plus, comanda postfix poate fi folosită cu același scop:
cuirass:~# postfix stop
postfix/postfix-script: stopping the Postfix mail system
cuirass:~# postfix start
postfix/postfix-script: starting the Postfix mail system
cuirass:~# postfix reload
postfix/postfix-script: refreshing the Postfix mail system
```

Pachetul *Postfix* instalează alte programe utile pentru comanda și gestiunea serverului *Postfix* precum `newalisases` sau `postconf`.

8.5.3.1 Jurnalizare

Verificarea corectitudinii funcționării serverului *Postfix* se realizează, de obicei, prin inspecția fișierelor de jurnalizare. În mod obișnuit, fișierele de jurnalizare se găsesc în `/var/log/mail.*`. Există, de obicei, 4 fișiere corespunzătoare diferitelor niveluri de jurnalizare și eroare:

- `mail.log`
- `mail.info`
- `mail.err`
- `mail.warn`

Aceste fișiere pot fi inspectate pentru a verifica ce mesaje sunt transmise sau pentru a verifica erorile apărute la transmisia unui mesaj. Cele 4 fișiere corespund nivelurilor de jurnalizare diferite: erori (`.err`), avertismente (`.warn`), informații (`.info`), operații jurnalizate (`.log`).

8.5.3.2 Alte comenzi utile

- `mailq` - afișează informații despre mesajele din coada de mesaje (ID, dimensiune, sursă, destinație, motivul pentru care nu a fost livrat - dacă este cazul, etc.);
- `postsuper -d queue_id` - șterge mesajul identificat de `queue_id` din coada de mesaje; pentru ștergerea tuturor mesajelor se utilizează `postsuper -d ALL`;
- `postqueue -f` - forțează retrimiterea mesajelor din coadă.

8.5.4 Fișiere de configurare

Fișierele de configurare ale *Postfix*-ului se găsesc în `/etc/postfix`, cele mai importante fiind `/etc/postfix/main.cf` și `/etc/postfix/master.cf`. Fișierul principal de configurare este `/etc/postfix/main.cf`. După fiecare modificare a acestor fișiere este necesară repornirea serverului:

```
# /etc/init.d/postfix reload
```

sau

```
# postfix reload
```

Material cu drept de autor

Fișierul principal de configurare conține directive în forma `nume = valoare` și afectează funcționarea serviciului:

```
cuirass:~# tail -3 /etc/postfix/main.cf
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all
```

Fișierul `master.cf` definește funcționarea *daemon*-ului master, procesul care coordonează pornirea celorlalte procese Postfix:

```
# service type  private unpriv  chroot  wakeup  maxproc command + args
#               (yes)   (yes)   (yes)   (never) (100)
# ==========================================================================
smtp      inet  n       -       -       -       -       smtpd
[...]
pickup    fifo  n       -       -       60      1       pickup
cleanup   unix  n       -       -       -       0       cleanup
```

8.5.4.1 postconf

Alterarea fișierului `/etc/postfix/main.cf` poate fi realizată prin intermediul unui editor, dar se recomandă folosirea comenzii `postconf`. Comanda `postconf` poate fi folosită pentru verificarea fișierului principal de configurare, pentru vizualizarea directivelor de configurare sau pentru modificarea unei directive. Directivele care nu sunt configurate în `/etc/postfix/main.cf` au o valoare implicită.

Afișarea configurației actuale se realizează prin rularea fără argumente a comenzii `postconf`:

```
cuirass:~# postconf | head -3
2bounce_notice_recipient = postmaster
access_map_reject_code = 554
address_verify_default_transport = $default_transport
```

Editarea unei directive se realizează prin intermediul argumentului `-e`:

```
cuirass:~# postconf mailbox_size_limit
mailbox_size_limit = 0
cuirass:~# postconf -e mailbox_size_limit=30000000
cuirass:~# postconf mailbox_size_limit
mailbox_size_limit = 30000000
```

8.5.5 Configurare de bază

Directivele importante din fișierul principal de configurare afectează parametri precum nume de domeniu, interfețe și rețele active, suport TLS și SASL, utilizatori virtuali și căsuțe poștale virtuale.

8.5.5.1 Configurare domenii

Directivele de bază pentru configurarea numelor de domeniu sunt `myorigin` și `mydestination`.

Directiva `myorigin` specifică domeniul sursă pentru mesajele transmise de pe sistemul local:

```
cuirass:~# postconf myorigin
myorigin = /etc/mailname
cuirass:~# cat /etc/mailname
cuirass.localdomain
```

Spre exemplu, în cazul folosirii comenzii `mailx`, domeniul specificat de `myorigin` va fi atașat la numele utilizatorului care a folosit comanda. În cazul de mai sus, dacă utilizatorul `ana` transmite un mesaj folosind `mailx`, expeditorul va fi `ana@cuirass.localdomain`.

Directiva `mydestination` specifică domeniile pentru care mesajele sunt păstrate local. Mesajele destinate utilizatorului `test_user@test_domain.com` vor fi livrate utilizatorului local

Material cu drept de autor

`test_user` dacă domeniul `test_domain` este prezent în cadrul directivei `mydestination`. Un server *Postfix* poate asigura găzduire virtuală pentru mai multe domenii prin adăugarea acestora în directiva `mydestination`. În listingul de mai jos se configurează *Postfix* pentru a accepta mesaje şi pentru domeniul `test.cs.pub.ro`:

```
cuirass:~# postconf mydestination
mydestination = cuirass.localdomain, localhost.localdomain, , localhost
cuirass:~# postconf -e 'mydestination = cuirass.localdomain, localhost.localdomain, ,
localhost, test.cs.pub.ro'
cuirass:~# postconf mydestination
mydestination = cuirass.localdomain, localhost.localdomain, , localhost, test.cs.pub.ro
```

Configuraţia este completă dacă există şi o mapare între numele de domeniu şi adresa IP a sistemului pe care rulează *Postfix*. Aceasta se poate realiza printr-o configurare DNS sau, pentru testare locală, prin adăugarea unei mapări în fişierul `/etc/hosts`:

```
cuirass:~# cat /etc/hosts | grep test
172.16.68.128  test.cs.pub.ro
cuirass:~# ping -c 1 test.cs.pub.ro
PING test.cs.pub.ro (172.16.68.128) 56(84) bytes of data.
64 bytes from test.cs.pub.ro (172.16.68.128): icmp_seq=1 ttl=64 time=1.20 ms
```

În această configuraţie, mesajele livrate către ana@localhost, ana@localhost.localdomain, ana@cuirass.localdomain, ana@test.cs.pub.ro vor ajunge în căsuţa poştală a utilizatorului ana.

8.5.5.2 Configurare interfeţe şi porturi

Directiva inet_interfaces precizează interfeţele pe care Postfix ascultă conexiuni. Implicit, Postfix ascultă conexiuni pe toate interfeţele:

```
cuirass:~# postconf inet_interfaces
inet_interfaces = all
```

Postfix poate fi configurat să asculte conexiuni doar pe o interfaţă sau pe câteva interfeţe:

```
cuirass:~# postconf inet_interfaces
inet_interfaces = all
cuirass:~# netstat -tlnp | grep 25
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN      4926/master
cuirass:~# postconf -e 'inet_interfaces = 127.0.0.1, 172.16.68.128'
cuirass:~# /etc/init.d/postfix restart
Stopping Postfix Mail Transport Agent: postfix.
Starting Postfix Mail Transport Agent: postfix.
cuirass:~# netstat -tlnp | grep 25
tcp        0      0 172.16.68.128:25        0.0.0.0:*               LISTEN      5041/master
tcp        0      0 127.0.0.1:25
```

Configurarea unui port pe care *Postfix* să asculte conexiuni se realizează prin intermediul fişierului de configurare pentru *daemon*-ul master. Astfel, dacă se doreşte ca *Postfix* să asculte conexiuni şi pe portul 2525, se adaugă linia de mai jos la fişierul `/etc/postfix/master.cf`:

```
2525      inet  n       -       -       -       -       smtpd
```

După repornire, *Postfix* va asculta conexiuni şi pe portul 2525:

```
cuirass:~# /etc/init.d/postfix restart
Stopping Postfix Mail Transport Agent: postfix.
Starting Postfix Mail Transport Agent: postfix.
cuirass:~# netstat -tlnp | grep 25
tcp        0      0 0.0.0.0:25              0.0.0.0:*               LISTEN      4926/master
tcp        0      0 0.0.0.0:2525            0.0.0.0:*               LISTEN      4926/master
```

Pentru a nu acţiona ca *open relay*, *Postfix* acceptă transmiterea de mesaje către domenii pentru care nu este destinatar doar de la staţii din anumite reţele. Precizarea acestor reţele se realizează prin intermediul directivei `mynetworks`. Mesajele sosite de la staţii din aceste reţele vor fi livrate indiferent de destinaţie. Cele sosite de la staţii din alte reţele vor fi livrate local dacă serverul este responsabil de domeniul destinaţie (domeniul este asociat directivei `mynetworks`) altfel vor fi respinse.

În configuraţia de mai jos:

Material cu drept de autor

```
cuirass:~# postconf mynetworks
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
```

Postfix va acționa ca *relay* doar pentru mesajele trimise de pe stația locală. Mesaje sosite din rețea nu vor fi livrate domeniilor de care serverul nu este responsabil.

În listarea de mai jos se încearcă trimiterea unui mesaj către domeniul `gmail.com` prin conectare pe interfața 172.16.68.128. Specificarea destinatarului eșuează cu precizarea mesajului *„Relay access denied"*

```
cuirass:~# telnet 172.16.68.128 25
Trying 172.16.68.128...
Connected to 172.16.68.128.
Escape character is '^]'.
220 cuirass.localdomain ESMTP Postfix (2.5.5)
EHLO localhost
[...]
MAIL FROM: ana
250 2.1.0 Ok
RCPT TO: razvand@gmail.com
554 5.7.1 <razvand@gmail.com>: Relay access denied
```

8.5.5.3 Configurare relaying

Dacă se dorește configurarea de domenii către care mesajul să fie transmis indiferent de sursă, se folosește directiva `relay_domains`. Directiva specifică domeniile pentru care se va face *relay*, pe lângă domeniile din *mydestination*.

În exemplul de mai jos se specifică domeniul `gmail.com` ca domeniu pentru care se face *relay*:

```
cuirass:~# postconf relay_domains
relay_domains = $mydestination
cuirass:~# postconf -e 'relay_domains = gmail.com'
```

Tabelul de mai jos explică modul în care se realizează transmiterea mesajelor în diverse situații:

Rețea sursă în mynetworks	Domeniu destinație în mydestination	Domeniu destinație în relay_domains	Efect
DA	Nu contează	Nu contează	Transmitere mesaj
NU	DA	Nu contează	Livrare locală
NU	NU	DA	Transmitere mesaj
NU	NU	NU	Mesaj respins

8.5.6 Configurare utilizatori virtuali și căsuțe poștale virtuale

Serverul *Postfix* oferă facilități de găzduire virtuală, adică poate fi configurat ca *Mail Exchange* pentru mai multe domenii. Configurarea se realizează prin intermediul directivei `mydestination`.

În plus, *Postfix* permite configurarea de utilizatori virtuali. Astfel de utilizatori nu există în sistem și mesajele către aceștia sunt livrate fie unor utilizatori locali fie unor utilizatori de pe alt sistem. Pot fi configurate, de asemenea, alias-uri pentru utilizatorii locali astfel încât mesajele livrate unui utilizator local să fie redirectate altui utilizator sau unui alt sistem.

În fine, *Postfix* permite configurarea de utilizatori virtuali care folosesc căsuțe poștale virtuale. Altfel spus, pot fi configurați utilizatori care nu au cont în sistem și li se pot asocia intrări specializate în sistemul local de fișiere reprezentând căsuțele poștale. Un utilizator virtual nu are cont în sistem și nici nu are un director `home` asociat. Are însă o intrare în sistemul local de fișiere reprezentând căsuța poștală și poate primi și transmite mesaje.

Material cu drept de autor

8.5.6.1 Configurare aliasuri

Configurarea cea mai simplă de alias-uri în *Postfix* se realizează prin interfaţa compatibilă *Sendmail*. Fişierul pentru configurarea de alias-uri este `/etc/aliases`. După editarea acestui fişier activarea alias-urilor se realizează prin intermediul comenzii `newaliases`.

Fişierul are o structură de forma `alias: adrese_finale`. Adresele finale unde va fi livrat mesajul pot fi separate prin spaţiu sau prin virgulă. Dacă se doreşte ca mesajele trimise către elena să fie livrate utilizatorului `florin` şi unei adrese externe se va adăuga în fişierul `/etc/aliases` o linie de forma:

```
elena: florin extern@example.org
```

După care se va rula comanda `newaliases`:

```
# newaliases
```

Utilizatorul `elena` poate să nu existe în sistem.

De multe ori un utilizator nu doreşte folosirea contului de pe un sistem ci redirectarea mesajelor către un alt cont. Acest lucru se realizează cu ajutorul fişierului `.forward` din `home`-ul utilizatorului. Astfel, dacă un utilizator doreşte redirectarea mesajelor din contul său către contul `extern@example.org`, va crea un fişier `.forward` în care va adăuga acea adresă:

```
$ cat .forward
extern@example.org
```

Orice adresă adăugată ulterior va însemna transmiterea mesajului şi către acea adresă.

8.5.6.2 Configurare utilizatori virtuali pentru domenii multiple

În cazul folosirii suportului de găzduire virtuală, un server *Postfix* va servi mai multe domenii. De obicei se va dori ca mesajele transmise către fiecare domeniu să ajungă altundeva. Astfel, dacă un server *Postfix* serveşte domeniiile `alfa.com` şi `beta.com`, o cerinţă poate fi ca mesajele către `info@alfa.com` să fie livrate utilizatorului local `florin` iar mesajele livrate către `info@beta.com` să fie livrate adresei `extern@example.org`.

Interfaţa de alias-uri *Sendmail* nu permite ca utilizatorul virtual/alias-ul să conţină şi un nume de domeniu. Astfel, nu se poate face deosebirea între utilizatorul `info@beta.com` şi `info@alfa.com`. Pentru aceasta se folosesc directive specializate *Postfix* pentru domenii virtuale.

Directiva `virtual_alias_domains` specifică domeniile virtuale pe care le serveşte *Postfix*. Domeniile precizate aici nu trebuie să se regăsească în cadrul directivei `mydestination`:

```
cuirass:~# postconf -e 'virtual_alias_domains = alfa.com, beta.com'
```

Fişierul care va conţine alias-urile este precizat prin intermediul directivei `virtual_alias_maps`:

```
cuirass:~# postconf -e 'virtual_alias_maps = hash:/etc/postfix/virtual_alias'
```

Intrările în fişierul de alias-uri sunt în forma `alias destinaţie`:

```
cuirass:~# cat /etc/postfix/virtual_alias
info@alfa.com          elena
info@beta.com          florin
contact@alfa.com       extern@example.org
```

În configuraţia de mai sus, mesajele destinate către `info@alfa.com` vor fi livrate utilizatorului local `elena`, iar cele destinate `info@beta.com` utilizatorului local `florin`. De asemenea, mesajele destinate `contact@alfa.com` vor fi livrate contului `extern@example.org`.

Material cu drept de autor

După completarea fișierului de alias-uri, activarea se realizează prin intermediul comenzii `postmap`:

```
cuirass:~# postmap /etc/postfix/virtual_alias
```

Dacă utilizatorii `elena` și `florin` doresc ca adresele sursă pentru mesajele trimise să fie `info@alfa.com`, respectiv `info@beta.com` se folosește directiva `canonical_maps`. Se pot folosi directivele `sender_canonical_maps`, respectiv `recipient_canonical_maps` dacă se soreşte modificarea adreselor sursă, respectiv destinație:

```
cuirass:~# postconf -e 'sender_canonical_maps = hash:/etc/postfix/canonical'
cuirass:~# cat /etc/postfix/canonical
ana     info@alfa.com
bogdan  info@beta.com
cuirass:~#    postconf    -e    'local_header_rewrite_clients    =    permit_mynetworks,
permit_sasl_authenticated'
cuirass:~# postmap /etc/postfix/canonical
```

Directiva `local_header_rewrite_clients` selectează clienții pentru care se va realiza suprascrierea adresei sursă. Această opțiune este utilă pentru situația în care se dorește substituția conturilor din sistem cu adrese de forma `Nume.Prenume`.

8.5.6.3 Configurare căsuțe poștale virtuale

Un server de e-mail cu foarte mulți utilizatori ar necesita existența unui număr foarte mare de conturi. Dincolo de problemele de scalabilitate, gestiunea utilizatorilor devine greoaie. Soluția este folosirea de căsuțe poștale virtuale. Utilizatorii vor fi utilizatori virtuali iar căsuța poștală va fi o intrare specializată în sistemul local de fișiere.

Pentru precizarea domeniilor care folosesc căsuțe poștale virtuale se folosește directiva `virtual_mailbox_domains`:

```
cuirass:~# postconf -e 'virtual_mailbox_domains = gamma.com'
```

De obicei se va asocia un director pentru fiecare domeniu virtual. Spre exemplu `/var/mail/vhosts/gamma.com` pentru `gamma.com`. Directiva `virtual_mailbox_base` specifică directorul de bază pentru domeniile virtuale:

```
cuirass:~# postconf -e 'virtual_mailbox_base = /var/mail/vhosts'
```

Ca până acum, trebuie precizat fișierul ce va conține mapările între adresa de e-mail și intrarea în sistemul de fișiere reprezentând căsuța poștală virtuală:

```
cuirass:~# postconf -e 'virtual_mailbox_maps = hash:/etc/postfix/virtual'
cuirass:~# cat /etc/postfix/virtual
info@gamma.com gamma.com/info/
cuirass:~# postmap /etc/postfix/virtual
```

În cadrul mapării, intrarea asociată căsuței poștale virtuale este relativă la `virtual_maibox_base`. În cazul în care căsuța poștală este în format *Maildir* se folosește un caracter `/` (slash) la sfârșit. Căsuța poștală în format *Maildir* se creează cu ajutorul utilitarului `maildirmake`:

```
cuirass:~# mkdir -p /var/mail/vhosts/gamma.com
cuirass:~# cd /var/mail/vhosts/gamma.com/
cuirass:/var/mail/vhosts/gamma.com# maildirmake info
cuirass:/var/mail/vhosts/gamma.com# ls -l
total 4
drwx------ 5 root mail 4096 2008-09-17 01:24 info
```

După cum se observă, deținătorul căsuței poștale virtuale este `root`. Căsuța poștală virtuală va fi accesată și de serverul POP3/IMAP și trebuie configurat un utilizator comun atât pentru livrare (*Postfix*) cât și pentru acces (POP3/IMAP):

```
cuirass:/var/mail# useradd -g mail vmail
cuirass:/var/mail# id vmail
uid=1005(vmail) gid=8(mail) groups=8(mail)
```

Material cu drept de autor

```
cuirass:/var/mail# chown -R vmail:mail /var/mail/vhosts
```

Acest utilizator trebuie precizat şi în directorul principal de configurare:

```
cuirass:/var/mail# postconf -e 'virtual_uid_maps = static:1005'
cuirass:/var/mail# postconf -e 'virtual_gid_maps = static:8'
cuirass:/var/mail# postfix reload
postfix/postfix-script: refreshing the Postfix mail system
```

Mesajele transmise către `info@gamma.com` vor fi stocate în căsuţa poştală `/var/mail/vhosts/gamma.com/info/` în format *Maildir*. De aici vor putea fi citite prin configurarea corespunzătoare a unui server POP3/IMAP.

8.5.7 Configurare suport SASL şi TLS

8.5.7.1 Suport TLS

Ultimele versiuni de pachete postfix (> 2.3) oferă suport implicit pentru TLS. Astfel, fişierul principal de configurare conţine, în urma instalării, directive specifice pentru suport TLS:

```
cuirass:/etc/courier# cat /etc/postfix/main.cf | grep tls
smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_use_tls=yes
smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
```

Certificatul folosit pentru autentificare este generat automat la instalare. Certificatul nu este semnat, însă, de o autoritate de certificare şi va furniza un avertisment în momentul conectării clientului.

8.5.7.2 Suport SASL

Pentru autentificare folosind SASL trebuie instalată o serie de pachete specifice:

```
cuirass:~# apt-get install libsasl2 sasl2-bin libsasl2-modules
```

În continuare trebuie activată autentificarea folosin SASL:

```
cuirass:~# postconf -e 'smtpd_sasl_local_domain ='
cuirass:~# postconf -e 'smtpd_sasl_auth_enable = yes'
cuirass:~# postconf -e 'smtpd_sasl_security_options = noanonymous'
cuirass:~# postconf -e 'broken_sasl_auth_clients = yes'
cuirass:~# postconf -e 'smtpd_recipient_restrictions =
permit_sasl_authenticated,permit_mynetworks,reject_unauth_destination'
```

În plus, în fişierul `/etc/postfix/sasl/smtpd.conf` se specifică metoda de autentificare:

```
cuirass:~# echo 'pwcheck_method: saslauthd' >> /etc/postfix/sasl/smtpd.conf
cuirass:~# echo 'mech_list: plain login' >> /etc/postfix/sasl/smtpd.conf
```

În ultimă fază trebuie activat *daemonul* `saslauthd`. Întrucât *Postfix* rulează într-un `jail chroot`, `saslauthd` trebuie configurat corespunzător. Fişierul de configurare pentru `saslauthd` este `/etc/default/saslauthd`. Se creează directorul `/var/spool/postfix/var/run/saslauthd`:

```
cuirass:~# mkdir -p /var/spool/postfix/var/run/saslauthd
```

şi se configurează corespunzător serviciul de autentificare prin modificarea liniei:

```
OPTIONS="-c -m var/run/saslauthd"
```

în

```
OPTIONS="-c -m /var/spool/postfix/var/run/saslauthd"
```

De obicei se va activa pornirea automată a `saslauthd`:

```
START=yes
```

Material cu drept de autor

În absenţa unor mecanisme de verificare locală a utilizatorului, se recomandă folosirea mecanismului `shadow` care foloseşte fişierul local pentru autentificarea utilizatorilor (`/etc/shadow`):

```
MECHANISMS="shadow"
```

Se porneşte *daemonul* de autentificare:

```
cuirass:~# /etc/init.d/saslauthd start
Starting SASL Authentication Daemon: saslauthd.
```

8.5.7.2.1 Pachete noi (Debian Lenny)

În distribuţiile recente, folosirea `saslauthd` este condiţionată de reconfigurarea drepturilor şi apartenenţelor:

```
cuirass:~# dpkg-statoverride --add root sasl 710 /var/spool/postfix/var/run/saslauthd
cuirass:~# adduser postfix sasl
Adding user `postfix' to group `sasl' ...
Adding user postfix to group sasl
Done.
```

Ulterior se reporneşte *daemonul* `saslauthd` şi *Postfix*:

```
cuirass:~# /etc/init.d/saslauthd restart
Stopping SASL Authentication Daemon: saslauthd.
Starting SASL Authentication Daemon: saslauthd.
cuirass:~# /etc/init.d/postfix restart
Stopping Postfix Mail Transport Agent: postfix.
Starting Postfix Mail Transport Agent: postfix.
```

8.5.7.3 Verificare

Pentru verificarea suportului SASL şi TLS se foloseşte comanda SMTP EHLO:

```
cuirass:~# telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 cuirass.localdomain ESMTP Postfix (2.5.5)
EHLO localhost
250-cuirass.localdomain
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-STARTTLS
250-AUTH LOGIN PLAIN
250-AUTH=LOGIN PLAIN
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
```

Prezenţa liniilor STARTTLS şi AUTH LOGIN PLAIN înseamnă suport valid TLS şi SASL pentru Postfix. Pentru folosirea acestui suport clienţii de e-mail trebuie configuraţi corespunzător.

8.6 MDA

În general, serverele de e-mail despre care s-a discutat până acum includ o componentă care se ocupă de livrarea locală a mesajelor. Există, însă, situaţii în care se doreşte mai multă flexibilitate în livrarea mesajelor, de la aranjarea mesajelor în cutii poştale speciale în funcţie de proveniența lor, până la activarea de filtre *antispam*. Spre exemplu, *Sendmail* nu oferă suport pentru căsuţe poştale în format *Maildir*. În această situaţie trebuie folosit un MDA pentru livrarea mesajelor către căsuţele poştale în format *Maildir*.

8.6.1 Procmail

Procmail formează, împreună cu *Maildrop*, cele mai cunoscute două MDA-uri. *Procmail* este, de obicei, instalat implicit pe distribuţiile *Debian-based*. Altfel, se poate instala folosind `apt-get`:

Material cu drept de autor

```
# apt-get install procmail
```

8.6.1.1 Interacţiunea cu Procmail

Deşi *Procmail* poate rula ca aplicaţie de sine stătătoare, este invocat de obicei de MTA. Pentru a configura *Postfix* să folosească *Procmail* se adaugă linia:

```
mailbox_command = /path/to/procmail -a $EXTENSION
```

Folosirea liniei de mai sus dezactivează configurarea oferită de `home_mailbox`.

8.6.1.2 Configurarea Procmail

Configurarea *Procmail* se face prin intermediul fişierului global de configurare `/etc/procmailrc` (dacă există) şi a unui fişier local de configurare: `$HOME/.procmailrc`.

În momentul recepţionării unui mesaj, *Procmail* va începe procesarea fişierului `/etc/procmailrc` şi apoi a fişierului *.procmailrc* din `home`-ul utilizatorului. Un astfel de fişier este compus dintr-o parte de declaraţii şi o parte de reguli de filtrare (*recipes*).

Dacă se doreşte transmiterea globală a mesajelor către căsuţe poştale virtuale în format *Maildir* se configurează corespunzător fişierul global de configurare:

```
cuirass:~# cat /etc/procmailrc
DEFAULT=$HOME/Maildir/
MAILDIR=$HOME/Maildir
LOGFILE=/usr/local/proc.log
```

Mai jos este prezentat un exemplu simplu de fişier local de configurare

```
bogdan@cuirass:~$ cat .procmailrc
PATH=/bin:/usr/bin:/usr/local/bin
SHELL=/bin/bash
LOGFILE=$HOME/proclog
DEFAULT=$HOME/Maildir/
MAILDIR=$HOME/Maildir
:0:
* ^Subject: .*test.*
$MAILDIR/.Test/
# catch-all rule
:0
$HOME/Maildir/
```

8.6.1.2.1 Directive de configurare

Directivele sunt definite în forma `NUME=valoare`. În exemplul de mai sus, se definesc directivele DEFAULT şi MAILDIR pentru a preciza intrarea în sistemul de fişiere asociată căsuţei poştale virtuale. Directiva LOGFILE precizează fişierul folosit pentru jurnalizare. În cazul în care nu se doreşte stocarea jurnalelor se poate folosi `/dev/null`. Directiva PATH este utilă în cazul în care se folosesc comenzi de filtrare externe şi, din considerente de eficienţă, nu se doreşte folosirea căii complete către comandă.

8.6.1.2.2 Reguli de filtrare

Orice regulă începe cu caracterele `:0` urmate de unul sau mai multe câmpuri de control şi, opţional, de specificarea unui fişier pentru *lock* utilizat pentru a preveni modificarea simultană a locaţiei unde va fi salvat mesajul. Pe liniile următoare se pot specifica una sau mai multe condiţii urmate de o linie ce reprezintă acţiunea aplicată dacă toate condiţiile anterioare sunt valabile. Formal, formatul este:

```
:0 [flags] [: [lock-file] ]
zero or more conditions
one action line
```

În exemplul de mai sus, mesajele al căror subiect conţin şirul `test` sunt stocate în directorul `.Test`. Directorul `.Test` corespunde unui director `Test` vizibil din clientul de e-mail

Material cu drept de autor

în cazul folosirii unei căsuţe poştale în format *Maildir*. Ultima regulă (*catch-all*), stochează mesajele în căsuţa poştală în format *Maildir*. Folosirea caracterului două puncte în cazul primei reguli înseamnă folosirea fişierului implicit de *locking*.

Cele mai importante câmpuri de control sunt:

- **H** - verificarea condiţiilor se face în antetul mesajului (H = *header*); dacă nu se specifică nimic, acest câmp este configurat implicit;
- **B** - verificarea condiţiilor se face în corpul mesajului;
- **D** - *case sensitive* (implicit nu se face distincţie între literele mari şi mici);
- **c** - se generează o copie a mesajului;
- **w** - se aşteaptă ca programul sau filtrul să se termine şi verifică valoarea de ieşire a acestuia (codul de retur); dacă nu s-a terminat cu succes, atunci filtrul nu este aplicat.

Condiţiile sunt formate din expresii regulate la care se adaugă câţiva operatori speciali cum ar fi „!" pentru inversarea condiţiei sau „<" respectiv „>" pentru a compara dimensiunea mesajului cu anumite limite.

Acţiunile ce pot fi aplicate sunt:

- o intrare în sistemul local de fişiere unde va fi stocat mesajul;
- **!** - trimite mesajul la adresa specificată;
- **|** - porneşte un program specificat;
- **{ }** - specifică un bloc în interiorul căruia putem specifica alte reguli.

8.6.1.3 Câteva exemple

În continuare sunt prezentate câteva exemple de reguli de filtrare. Mai multe exemple sunt descrise în pagina de manual `procmailex(5)`.

- se salvează în cutia poştală `mail/stiri/agora` toate mesajele al căror câmp **From** conţine expresia `agnews@agora.ro`:

```
:0
* From: .*agnews@agora.ro
mail/stiri/agora
```

- se trimit toate mesajele către `rc@cs.pub.ro`:

```
:0
! rc@cs.pub.ro
```

- se generează o copie a mesajului şi se trimite către `rc@cs.pub.ro`; mesajul original va fi verificat în continuare cu regulile următoare:

```
:0c
! rc@cs.pub.ro
```

- pentru toate mesajele cu subiectul **[humor]** se va genera o copie ce va fi trimisă la `rc@cs.pub.ro`, iar copia locală se va stoca într-o cutie poştală definită, în fişierul `/liste/humor`:

```
:0
* ^Subject:.*[humor]
{
    :0 c
    ! rc@cs.pub.ro
    :0
    mail/liste/humor
}
```

- se filtrează cu *SpamAssassin* toate mesajele cu dimensiunea mai mică de 256 kB:

```
:0fw
* < 262144
| /usr/bin/spamassassin -P
```

Material cu drept de autor

8.7 Servere de IMAP

Cele mai răspândite servere de IMAP pentru platforme Unix sunt *Courier IMAP Server*, *University of Washington imapd* şi *Cyrus IMAP Server* de la *Carnegie Mellon University*.

8.7.1 Courier IMAP Server

Courier IMAP Server este o componentă a serverului *Courier*. Nu oferă suport pentru formatul *mbox* ci doar pentru *Maildir*. Serverul implementează extensii ale acestui format, permiţând, spre exemplu, posibilitatea stabilirii unor limite (*quota*) sau structurarea ierarhică a *mailbox-urilor*. Serverul are trei avantaje importante:

- permite definirea de căsuţe poştale virtuale;
- are numeroase module de autentificare;
- oferă posibilitatea limitării numărului de conexiuni IMAP de la o anumită adresă IP.

O altă caracteristică este posibilitatea partajării căsuţelor poştale între mai mulţi utilizatori.

Courier IMAP Server a fost construit într-o manieră modularizată pentru a permite un consum minim de resurse.

8.7.1.1 Instalare

Instalarea serverului se realizează prin intermediul instalării pachetului `courier-imap`.

```
# apt-get install courier-imap
```

Singura întrebare care se pune la instalarea pachetului este dacă se doreşte crearea unei structuri de directoare pentru configurare care permite configurarea folosind o interfaţă web.

Instalarea pachetului şi a dependinţelor acestuia conduce la rularea unui server de IMAP (`imapd`) pe portul implicit (143) şi a unui set de utilitare specifice. Un utilitar important este `maildirmake` care permite crearea unui director în format *Maildir* reprezentând căsuţa poştală folosită pentru recepţionarea mesajelor.

8.7.1.2 Configurare

Fişierele de configurare pentru serverul de IMAP se găsesc în `/etc/courier/` şi sunt `imapd`, pentru configurarea funcţionalităţii serverului, şi `authdaemonrc`, pentru configurarea *daemon*-ului de autentificare.

Directivele din fişierul `imapd` sunt în formatul `NUME=valoare`. Aici se pot schimba adresa pe care ascultă serverul, portul, numărul de procese care pot fi deschise, numărul maxim de conexiuni, etc. De asemenea, aici se pot stabili alte valori pentru numele directoarelor implicite, prin alterarea directivelor respective. Astfel, dacă se doreşte schimbarea numelui directorului implicit de recepţie a mesajelor (`Maildir`), se va înlocui linia

```
MAILDIRPATH=Maildir
```

cu

```
MAILDIRPATH=Mymail
```

Comanda `makemaildir` este utilizată pentru crearea unui director cu formatul utilizat de *Courier* (`maildir`). O opţiune utilă în cadrul acestei comenzi este posibilitatea asocierii unei cote (`quota`). Mai jos sunt prezentate două exemple de utilizare. Comanda:

```
bogdan@cuirass:~$ ls
bogdan@cuirass:~$ maildirmake Maildir
bogdan@cuirass:~$ ls
Maildir
```

creează un director format *Maildir* în directorul de bază al utilizatorului; comanda:

Material cu drept de autor

```
$ maildirmake -q 10000000S $HOME/Maildir
```

creează un director format *Maildir* în directorul de bază al utilizatorului cu o cotă de 10 MB. Trebuie specificat faptul că numai anumite LDA-uri (*Courier Maildrop* şi *deliverquota*) vor ţine cont de cotele impuse într-o astfel de utilizare.

8.7.1.3 Utilizarea Courier IMAP cu Postfix

Serverul *Postfix* nu lucrează implicit cu formatul *Maildir*. Pentru aceasta va trebui modificat corespunzător fişierul de configurare `/etc/postfix/main.cf`, prin alterarea directivei `home_mailbox` şi ignorarea directivei `mailbox_command`:

```
cuirass:/etc/courier# postconf home_mailbox
home_mailbox =
cuirass:/etc/courier# postconf -e 'home_mailbox = Maildir/'
cuirass:/etc/courier# postconf home_mailbox
home_mailbox = Maildir/
cuirass:~# cat /etc/postfix/main.cf | grep command
#mailbox_command = procmail -a "$EXTENSION"
```

Nu trebuie omis caracterul / de la sfârşitul fişierul `Maildir`.

Anumite MTA-uri nu oferă suport pentru formatul *Maildir*. În această situaţie livrarea mesajelor trebuie delegată unui MDA precum *Procmail* sau *Maildrop*.

8.7.1.4 Suport SSL

Suportul IMAP peste SSL în cazul Courier IMAP este asigurat prin instalarea pachetului `courier-imap-ssl`:

```
cuirass:~# apt-get install courier-imap-ssl
```

În cadrul instalării pachetului se generează şi un certificat pentru autentificare în `/etc/courier/imapd.pem`.

Serverul IMAPS ascultă conexiuni pe portul 993:

```
cuirass:/etc/courier# netstat -tlnp | grep 993
tcp6       0      0 :::993                  :::*                    LISTEN      2173/couriertcpd
```

Fişierul `/etc/courier/imapd-ssl` este fişierul de configurare pentru serverul IMAPS.

8.7.1.5 Configurarea accesului la căsuţe poştale virtuale

Courier IMAP poate fi configurat pentru accesarea mesajelor din căsuţe poştale virtuale adică intrări specializate în sistemul de fişiere. Pentru aceasta trebuie configurat *daemonul* de autentificare – `authdaemond` prin intermediul fişierului `/etc/courier/authdaemonrc`.

În prima fază trebuie folosit modulul `authuserdb` care permite configurarea facilă a utilizatorilor virtuali. Acest lucru se realizează prin alterarea directivei de configurare `authmodulelist`:

```
authmodulelist="authuserdb authpam"
```

După configurare trebuie repornit *daemon*-ul de autentificare cu ajutorul scriptului `invoke-rc.d`:

```
cuirass:~# invoke-rc.d courier-authdaemon reload
Stopping Courier authentication services: authdaemond.
Starting Courier authentication services: authdaemond.
```

În continuare se adaugă utilizatorul `info` în baza de date de utilizatori şi se activează. Se presupune căsuţa poştală virtuală descrisă în secţiunea asociată *Postfix* [TODO] pentru contul `info@gamma.com`:

```
cuirass:~# userdb info set uid=1005 gid=8 home=/var/mail/vhosts/gamma.com/info
mail=/var/mail/vhosts/gamma.com/info
cuirass:~# userdbpw -md5 | userdb info set systempw
```

Material cu drept de autor

```
Password:
Reenter password:
cuirass:~# makeuserdb
```

În continuare clientul de e-mail se configurează pentru citirea mesajelor de pe contul `info@gamma.com`.

Pentru depistarea eventualelor erori de autentificare, se recomandă activarea opțiunii de jurnalizare a operației în fișierul de configurare `/etc/courier/authdaemonrc`:

```
DEBUG_LOGIN=1
```

8.8 Webmail

În mod frecvent, accesarea căsuțelor poștale se realizează utilizând diferiți clienți IMAP/POP3 instalați pe sistemul clientului, cum sunt, spre exemplu, *Mozilla Thunderbird*, *Outlook Express*, *Kmail*, *Evolution*, *Mutt*, etc. În multe situații se dorește posibilitatea accesării rapide a căsuței poștale de pe diferite sisteme fără a instala sau configura pe acestea un client de e-mail. Pentru asemenea situații există aplicații de accesare a căsuțelor poștale prin interfața web. Cele mai cunoscute, la momentul actual, sunt *Horde*, *SquirrelMail* și *RoundCube*.

8.9 Studii de caz

8.9.1 Comenzi SMTP. Transmiterea unui mesaj folosind SMTP

Se vor prezenta, în continuare, cele mai utilizate comenzi SMTP, fără a intra în detalii de sintaxă. Un mod simplu de a transmite un mesaj manual (fără utilizarea unui MUA) este conectarea la un server SMTP pe portul 25 utilizând **telnet**, ca în exemplul ce urmează:

```
root@MPLS-2:/home/mpls2# telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^]'.
220 localhost.localdomain ESMTP Postfix (Debian/GNU)
EHLO test
250-localhost.localdomain
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250 8BITMIME
MAIL FROM: mpls2@MPLS-2.cs.pub.ro
250 Ok
RCPT TO: razvand@gmail.com
250 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
hello,
un mesaj simplu.
.
250 Ok: queued as 0BC1A211F64
QUIT
221 Bye
Connection closed by foreign host.
```

HELO - comandă utilizată pentru identificarea serverului și a clientului SMTP. Clientul va iniția sesiunea printr-o astfel de comandă în care își anunță numele complet (FQDN - *Fully Qualified Domain Name*). Dacă serverul acceptă sesiunea, va răspunde cu codul 250 urmat de numele său.

MAIL FROM - este prima din comenzile utilizate pentru trimiterea efectivă a mesajului și are ca rol specificarea identității autorului mesajului. Dacă răspunsul serverului este OK (250) nu înseamnă ca mesajul va fi neapărat livrat, deoarece, în funcție de alte comenzi, se poate întoarce un cod de eroare.

RCPT TO - permite specificarea destinatarului mesajului. Dacă serverul recunoaște utilizatorul, atunci se va răspunde cu 250 OK; astfel se va returna un cod de eroare

corespunzător. Dacă se dorește ca mesajul sa ajungă la mai multe destinații se va folosi comanda **RCPT TO** de mai multe ori. De remarcat este faptul că standardul permite ca adresele specificate prin comenzile **MAIL FROM** și **RCPT TO** să fie diferite de cele specificate de câmpurile *From* și *To* din corpul mesajului. Dar, după cum s-a precizat, mesajele sunt livrate în funcție de informațiile transmise prin SMTP, nu de ceea ce apare în conținut.

DATA - permite specificarea corpului mesajului. Acesta are o succesiune de linii, succesiune încheiată cu caracterul „." plasat singur pe o linie. Conținutul său ar trebui să fie conform formatului precizat în RFC 822 (2822), adică să conțină câmpurile *Date, Subject, To, Cc, From*, dar nu este o cerință obligatorie impusă de standardul SMTP.

RSET - permite anularea tranzacției curente. Serverul va șterge *buffer*-ele alocate mesajului curent și va reveni în starea de după HELO. Conexiunea cu clientul nu este întreruptă.

VRFY - permite verificarea validității căsuței poștale pentru un utilizator specificat ca parametru.

EXPN - permite verificarea dacă o adresă specificată este o listă de discuții și, în caz afirmativ, afișează membrii listei respective.

NOOP - comandă de testare a conexiunii. Nu se efectuează nicio operație; serverul trebuie să răspundă cu 250 OK.

QUIT - comandă utilizată pentru întreruperea conexiunii cu serverul.

HELP - comandă de ajutor.

8.9.2 Comenzi POP3. Citirea unui mesaj folosind POP3

Printre cele mai utilizate comenzi POP3 sunt următoarele:

USER username/PASS password - sunt întotdeauna primele comenzi folosite după stabilirea conexiunii între clientul și serverul de POP3. Sunt folosite pentru autentificarea clientului la server. Dacă informațiile de autentificare sunt corecte și serverul reușește să obțină un *lock* asupra căsuței poștale (pentru a nu putea fi modificată de un alt client simultan), atunci întoarce răspuns pozitiv. Comanda **APOP** este opțională pentru serverele POP3 și a fost introdusă din cauza inconvenientului comenzii **PASS** de a trimite parola în clar.

Un mic exemplu de funcționare a protocolului POP3 (prin conectare la server pe portul 110 și utilizare `telnet`) este următorul:

```
mpls2@MPLS-2:~$ telnet localhost 110
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^]'.
+OK
USER: mpls2
-ERR
USER mpls2
+OK
PASS parola2
+OK
LIST
+OK
1 831
2 505
3 521
.
RETR 3
+OK
Return-Path: <mpls2@MPLS-2.cs.pub.ro>
X-Original-To: mpls2@localhost.localdomain
Delivered-To: mpls2@localhost.localdomain
Received: from test (localhost.localdomain [127.0.0.1])
        by localhost.localdomain (Postfix) with SMTP id 9695E211F64
        for <mpls2@localhost.localdomain>; Tue, 20 Sep 2005 12:37:35 +0300 (EEST)
Message-Id: <20050920093735.9695E211F64@localhost.localdomain>
Date: Tue, 20 Sep 2005 12:37:35 +0300 (EEST)
From: mpls2@MPLS-2.cs.pub.ro
To: undisclosed-recipients:;

yet another simple message
```

Material cu drept de autor

```
.
DELE 1
+OK
LIST
+OK
2 505
3 521
.
QUIT
+OK
Connection closed by foreign host.
```

STAT - solicită serverului informații despre căsuța poștală a utilizatorului. Un răspuns pozitiv din partea serverului este `+OK <numar_mesaje> <dimensiune_totala>`.

LIST - dacă această comandă este apelată fără niciun argument, serverul va afișa informații despre mesaje, câte o linie pentru fiecare. Comanda se poate apela și având ca argument un număr de mesaj, serverul returnând în acest caz doar linia corespunzătoare mesajului. Nu se poate invoca pentru mesajele marcate pentru ștergere.

RETR msg - afișează conținutul mesajului `msg`; orice comandă ulterioară cu referire la acest mesaj va genera o eroare.

DELE msg - marchează mesajul `msg` pentru ștergere; mesajele nu vor fi șterse efectiv decât la închiderea sesiunii.

NOOP - clientul nu solicită nicio acțiune, ci doar răspuns pozitiv din partea serverului.

RSET - anulează marcajul de ștergere pentru toate mesajele marcate.

QUIT - serverul va șterge toate mesajele marcate pentru ștergere și va returna *+OK* sau **-ERR** în funcție de reușita operației.

HELP - comandă de ajutor.

Întrebări

1. Care afirmații sunt adevărate în ceea ce privește sistemul de poștă electronică?

- *SMTP este protocolul utilizat între MTA (*Mail Transfer Agent*).*
- *SMTP este protocolul utilizat de către o aplicație de tip MUA (*Mail User Agent*) pentru a transfera mesajele de pe server.*
- *SMTP utilizează portul 25.*
- *SMTP nu este folosit pentru poștă electronică.*

2. Pentru un server SMTP, termenul de relaying reprezintă:

- *a primi un mesaj care nu este destinat unui utilizator din domeniile gestionate de server și a-l transmite mai departe către destinație.*
- *a primi un mesaj care este destinat unui utilizator din domeniile gestionate de server și a-l transmite mai departe către destinație.*
- *termenul de relaying are sens în contextul unui* Mail User Agent*, nu al unui server SMTP.*
- *niciunul din răspunsurile de mai sus.*

3. Protocolul IMAP permite (alegeți 2 variante):

- *gestiunea offline a mesajelor.*
- *structurarea pe directoare a mesajelor.*
- *transferul de mesaje între MTA (*Mail Transfer Agent*).*
- *accesarea doar a unei singure căsute poștale la un moment dat.*

4. Care din următoarele variante NU este un server de IMAP?

- *Postfix*
- *Courier*
- *Cyrus*
- *Dovecot*

5. Care din următoarele directive NU este o directivă Postfix?

- *home_mailbox*
- *sender_canonical_maps*
- *mynetworks*
- *mod_ssl*

6. Care din următoarele porturi NU este asociat serviciului de e-mail?

- *53*
- *993*
- *25*
- *587*

7. Care din următoarele afirmații despre Postfix este falsă?

- *este o aplicație monolitică*
- *poate gestiona domenii virtuale*
- *are suport pentru formatul Maildir*
- *are suport pentru căsuțe poștale virtuale*

Material cu drept de autor

8. Care este ordinea corectă a transmiterii şi accesării unui mesaj?

- *MUA, MTA, MTA, MDA, server IMAP, MUA*
- *MUA, MDA, MTA, MUA, MTA, server IMAP*
- *MUA, server IMAP, MTA, MDA, MTA*
- *MDA, MUA, MTA, server IMAP, MTA*

9. Ce reprezintă un domeniu virtual?

- *un domeniu ai cărui utilizatori nu există efectiv pe server.*
- *un domeniu pentru care nu se face* relay.
- *un nume alternativ la serverului de mail.*
- *niciuna din variantele de mai sus.*

10. Care este efectul următoarelor filtre în Procmail?

```
:0
! test1@cs.pub.ro

:0c
* Subject:.*retele
! test2@cs.pub.ro
```

- *toate mesajele sunt forwardate către* ***test1@cs.pub.ro****.*
- *toate mesajele sunt forwardate către* ***test1@cs.pub.ro*** *şi în plus, acele mesaje care conţin în subiect cuvântul "retele" sunt forwardate către* ***test2@cs.pub.ro****.*
- *toate mesajele sunt forwardate către* ***test1@cs.pub.ro*** *şi în plus, o copie a acelor mesaje care conţin în subiect cuvântul "retele" sunt forwardate către* ***test2@cs.pub.ro****.*
- *niciuna dintre variantele de mai sus.*

9 World Wide Web

"If you don't have an E-mail address, you're in the Netherworld. If you don't have your own World Wide Web page, you're a nobody."

Clifford Stoll

Ce se învaţă din acest capitol?

- Ce este World Wide Web-ul
- Care sunt tehnologiile fundamentale ale web-ului
- Cum funcţionează serviciul de web
- Configurarea serverului de web Apache2.2
- Configurarea IIS7 pe Windows

Cine este...

Sir Tim Berners-Lee *este cercetătorul creditat cu inventarea World Wide Web-ului. Pe 25 decembrie 1990 a realizat prima comunicaţie HTTP în Internet între un server şi un client. În prezent este directorul World Wide Web Consortium (W3C).*

Robert McCool *este autorul webserver-ului NCSA HTTPd, ulterior cunoscut sub numele Apache HTTP Server. A scris prima versiune ca student la Universitatea din Illinois unde a lucrat cu echipa iniţială a NCSA Mosaic (unul dintre primele browser-e web). A contribuit la specificaţia iniţială a Common Gateway Interface (CGI) care s-a dovedit a fi un element cheie în realizarea unui web dinamic. În prezent este dezvoltator la Yahoo!.*

Ward Cunningham *este creatorul primului software de wiki numit WikiWikiWeb (1994). Software-ul de wiki a fost folosit iniţial în interiorul companiei sale. A lucrat la Microsoft Corporation şi la Eclipse Foundation iar în prezent este CTO la compania AboutUs.*

World Wide Web-ul este un spaţiu de informaţie în care elementele de interes, cunoscute sub numele de resurse, sunt recunoscute prin utilizarea unor identificatori globali, denumiţi URI (*Uniform Resource Identifiers*). Termenul nu trebuie confundat cu Internetul; *web*-ul este de fapt un serviciu care acţionează deasupra Internetului.

9.1 Modul de funcţionare a Web-ului

Resursele obişnuite ale web-ului sunt denumite pagini web. Pentru accesarea unei pagini web sau a unei alte resurse se începe prin introducerea URL-ului asociat acelei pagini în *browser*, sau prin folosirea unui *hyperlink* către pagina respectivă. În mod evident, un pas anterior constă în rezolvarea numelui serverului din URL într-o adresa IP folosind DNS.

Următorul pas este transmiterea unei cereri HTTP către serverul web care funcţionează la adresa IP rezolvată. Se solicită astfel pagina web (sau resursa) prezentă în URL. O pagină web obişnuită este, în cea mai mare parte, un fişier text în format HTML. În urma solicitării, serverul web identifică resursa şi o transmite clientului. Clientul este chiar *browser*-ul.

În continuare, sarcina *browser*-ului este de a reda pagina descrisă de fişierele HTML, CSS şi alte fişiere, încorporând imagini, *link*-uri şi alte resurse după cum este necesar. Rezultatul este afişarea paginii solicitate în ecranul *browser*-ului.

Majoritatea paginilor web vor conţine *hyperlink*-uri către alte pagini, către fişiere care pot fi descărcate sau către alte resurse web. O astfel de colecţie de resurse interconectate prin intermediul *hyperlink*-urilor a fost denumită un *web de informaţie*. Posibilitatea accesării şi

Material cu drept de autor

folosirii acestor resurse în Internet a produs ceea ce Sir Tim Berners-Lee a denumit, la începutul anilor '90, World Wide Web.

Sir Tim Berners-Lee are meritul de a fi găsit soluția de succes care să structureze cantitatea vastă de informație din cadrul Internetului apărută ca urmare a extinderii acestuia la sfârșitul anilor '80 și începutul anilor '90. După multe tentative nereușite de a organiza aceste informații, Berners-Lee a găsit soluția care s-a impus, prin folosirea conceptului de *hypertext* la un loc cu Internetul. În acest proces el a dezvoltat un sistem de identificatori globali unici pentru resurse din Web: URI (*Uniform Resource Identifiers*).

9.1.1 Uniform Resource Locator (URL)

Un URL[1] reprezintă un format standardizat de adresare a resurselor de pe Internet. Un URL este de fapt un URI (*Uniform Resource Identifier*); altfel spuse, este un identificator al unei resurse. Fiind mai cunoscut, termenul de URL va fi folosit în continuare. URL-ul a fost o inovație fundamentală în istoria Internetului. Sintaxa a fost proiectată pentru a fi generică, extensibilă și capabilă să exprime adresele în orice set de caractere utilizând un subset limitat de caractere ASCII.

Un URL combină într-o adresă simplă cele patru elemente de bază necesare pentru localizarea unei resurse oriunde în cadrul Internetului:

- **protocolul** folosit în cadrul comunicației,
- **serverul (host)** cu care se comunică,
- **portul** de pe server folosit pentru conectare,
- **calea** către resursa de pe server (spre exemplu un nume de fișier).

Sintaxa folosită în cadrul URL-ului este `protocol://server:port/cale`. Dacă se dorește și autentificare, sintaxa are formatul `protocol://nume:parola@server:port/cale`. Ultima formă a sintaxei poate fi folosită, spre exemplu, la autentificarea pe site-uri FTP.

Un exemplu de URL este cel de mai jos:

```
http://www.samplesite.org:80/pub/search.html?search=world&num=10
```

În acest exemplu:

- `http` este protocolul;
- `www.samplesite.org` este serverul;
- `80` este portul folosit pentru conectarea la server (de vreme ce 80 este valoarea implicită pentru protocolul HTTP, această porțiune putea fi omisă);
- `/pub/search.html` este calea către resursă;
- `?search=world&num=10` este șirul de interogare (această parte este opțională).

În absența câmpului protocol din URL, într-un *browser* se folosește implicit HTTP. De asemenea, întrucât portul 80 este cel implicit, în mod normal nu este specificat. Drept urmare, un utilizator va introduce numai un URL parțial precum `www.samplesite.org/pub/search.html`. Totodată, pentru a confirma relevanța `www` ca serviciu în Internet, o înregistrare DNS cu cheia `samplesite.org` va avea, de obicei, asociată aceeași adresă IP ca înregistrarea pentru `www.samplesite.org`. În acest fel, o bună parte din *site*-urile web pot fi accesate fără specificarea `www` în fața numelui de domeniu.

9.1.2 Hypertext Transfer Protocol

HTTP reprezintă principala metodă de obținere a informației în World Wide Web. Scopul inițial a fost crearea unei modalități de publicare și transmitere de pagini HTML. Dezvoltarea HTTP a fost coordonată de *World Wide Web Consortium* și *Internet Engineering Task Force*, culminând cu publicarea unei serii de RFC-uri, cel mai notabil RFC 2616, care descrie HTTP/1.1,

[1] http://tools.ietf.org/html/rfc1738

Material cu drept de autor

versiunea HTTP utilizată în aceste momente. Versiunea HTTP/1.1 aduce mai multe îmbunătățiri și caracteristici noi, dar rămâne perfect compatibilă cu HTTP/1.0.

HTTP este un protocol de tip **întrebare/răspuns** între clienți și servere. Un client web (de obicei un *browser*), stabilește o conexiune TCP pe un port al unei stații (portul implicit este portul 80). Un server HTTP ascultă pe acel port și așteaptă din partea clientului transmiterea unei cereri de forma:

```
GET /cale/catre/resursa HTTP/1.0
```

urmată de un mesaj de tip MIME conținând un set de antete pentru descrierea cererii și un câmp opțional de date. Unele antete sunt opționale, pe când altele, precum `Host` sunt obligatorii (pentru HTTP/1.1). După primirea cererii, serverul transmite clientului un șir de tip răspuns, cum ar fi `200 OK`, și un mesaj reprezentând fișierul cerut sau un mesaj de eroare sau altă informație.

Răspunsul trimis de server începe cu un cod ce indică tipul răspunsului, încadrându-se în următoarele categorii:

- `1xx`: informare;
- `2xx`: succes;
- `3xx`: redirectare;
- `4xx`: mesaj de interogare eronat;
- `5xx`: eroare la nivelul serverului.

HTTP diferă de alte protocoale care folosesc TCP (cum este FTP) prin **încheierea conexiunii** după ce o anumită cerere a fost satisfăcută. Acest lucru face din HTTP protocolul ideal pentru World Wide Web, unde paginile au legături către alte pagini pe alte servere. Lipsa unei conexiuni persistente impune programatorilor web folosirea unor metode alternative pentru a reține starea conexiunii. Printre acestea se numără cookies, variabile ascunse (în *form*-uri web) sau sesiuni pe server.

O altă caracteristică importantă a HTTP este **lipsa securității comunicației**. Acest lucru a condus la apariția HTTPS. HTTPS este versiunea securizată a HTTP, utilizând SSL/TLS (*Secure Sockets Layer/Transport Layer Security*) pentru a proteja traficul. Protocolul folosește implicit portul 443. SSL, inițial creat pentru a proteja HTTP (dar folosit acum împreună cu alte protocoale), este potrivit comunicațiilor HTTP întrucât poate asigura protecție chiar dacă numai unul din membrii comunicației (serverul) este autentificat. Aceasta este situația obișnuită în cazul tranzacțiilor HTTP pe Internet.

9.1.3 HyperText Markup Language

HTML face parte din categoria limbajelor descriptive (*markup languages*) și este folosit pentru crearea de pagini web și alte informații care să poată fi redate într-un *browser* web. HTML este folosit pentru a structura informația, descriind anumite porțiuni de text ca antete, paragrafe, liste, etc., și poate fi folosit pentru a defini semantica unui document.

Inițial definit de Sir Tim Berners Lee și apoi dezvoltat de IETF cu o sintaxă SGML simplificată, HTML este astăzi un standard internațional. Specificația HTML este menținută de *World Wide Web Consortium* (W3C).

Primele versiuni de HTML erau definite cu reguli sintactice destul de permisive, aspect care a ajutat la adoptarea sa către cei care nu aveau familiaritate cu publicarea web. *Browser*-ele realizau diverse presupuneri despre intenția acestora și continuau cu redarea paginii respective. De-a lungul timpului, în cadrul standardelor oficiale, a apărut intenția de a crea o sintaxă de limbaj din ce în ce mai strictă. Cu toate acestea, *browser*-ele continuă să redea pagini care sunt departe de un format HTML valid.

Material cu drept de autor

Versiunea curentă a specificației HTML este HTML 4.01. W3C a intenționat înlocuirea acestuia cu XHTML, care aplică regulile stricte ale XML în HTML. Adoptarea XHTML se realizează într-un ritm mai puțin rapid, drept pentru care W3C dezvoltă versiunea 5 a HTML.

Tipurile de marcaje existente în HTML sunt:

- marcaj **structural** - descrie scopul textului; spre exemplu `<h2>Section</h2>` direcționează browser-ul să redea `Section` ca un antet de nivel doi;
- marcaj **de prezentare** - descrie modul în care apare textul; de exemplu, `<b>boldface</b>` va reda `boldface` în text îngroșat;
- marcaj **hypertext** - leagă părți ale documentului către alte documente; spre exemplu, `<a href="http://en.wikipedia.org">Wiki</a>`, va reda șirul `Wiki` ca un hyperlink către URL-ul specificat.

9.1.4 Clienți web

Ca majoritatea serviciilor din Internet, comunicația prin HTTP se realizează folosind paradigma client-server. Clienții web sunt cunoscuți sub numele de **navigatoare web** (*browser-e*). Browser-ul este o aplicație care permite utilizatorului afișarea și interacțiunea cu documente HTML găzduite pe anumite servere web sau menținute în cadrul unui sistem de fișiere. Printre cele mai cunoscute browser-e se numără *Microsoft Internet Explorer*, *Mozilla Firefox*, *Opera* și *Safari*. Un *browser* este cel mai cunoscut tip de *user agent*.

Browser-ele web comunică cu serverele web folosind HTTP pentru obținerea de pagini web. HTTP permite *browser*-elor să solicite informații serverelor web și să obțină pagini web de la acestea. Paginile sunt localizate prin intermediul URL-ului. Formatul unei pagini web este de obicei cod HTML care este identificat în protocolul HTTP prin folosirea unui *MIME content type*.

Browser-ele sunt renumite datorită „războiului navigatoarelor" (*browser wars*), competiție legată de dominarea pieței *browser*-elor. Termenul este folosit pentru două perioade de timp: lupta dintre *Internet Explorer* și *Netscape Navigator*, la sfârșitul anilor '90, și creșterea popularității *Mozilla Firefox*, în detrimentul *Internet Explorer*, începând cu anul 2004. Aceste *războaie* se răsfrâng și asupra utilizatorilor, care preferă anumite *browser*-e în fața altora. În momentul scrierii acestei cărți, *Internet Explorer* domină piața *browser*-elor cu o pondere de 50%, urmat de *Mozilla Firefox* cu 45%.

Sistemele Unix prezintă și *browser*-e în linie de comandă utile în cazul testării rapide a unui site sau în absența interfeței grafice. Printre acestea se numără `lynx`, `links`, `elinks`, `w3m`.

9.1.5 Servere web

Un **server web** (server HTTP) este un serviciu/daemon care are rolul de a furniza documente clienților web. Clientul web se va conecta la server și îi va solicita acestuia o resursă necesară.

Deși aplicațiile care implementează un server web diferă în detaliu, conțin aceleași caracteristici de bază. Fiecare server web acceptă o cerere HTTP din rețea sau din Internet și furnizează un răspuns HTTP către solicitant. În general, răspunsul conține text HTML, dar poate fi un fișier text, o imagine, sau alt tip de document.

Cele mai cunoscute servere web sunt:

- Apache HTTP Server, de la Apache Software Foundation
- Internet Information Services (IIS), de la Microsoft
- Google Web Server de la Google
- Sun Java Web Server, de la Sun Microsystems
- Zeus Web Server, de la Zeus Technology

Material cu drept de autor

9.1.5.1 Funcţionarea unui server web

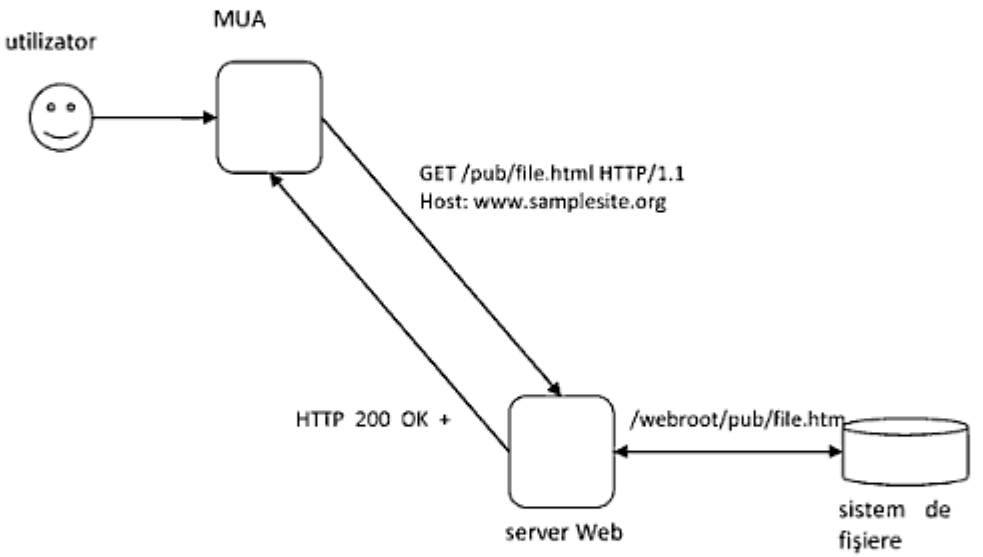
Serverele web translatează componenta **cale** *(path)* din cadrul unui URL într-o resursă din sistemul local de fişiere. Calea specificată în URL de către client este relativă la **directorul rădăcină** al serverului web (*webroot*).

Spre exemplu, se presupune situaţia în care un utilizator foloseşte URL-ul `http://www.samplesite.org/pub/file.html`. După introducerea acestui URL în bara de adrese a browser-ului, acesta îl translatează într-o conexiune către `www.samplesite.org` cu următoarea cerere HTTP 1.1:

```
GET /pub/file.html HTTP/1.1
Host: www.samplesite.org
```

Serverul web care rulează pe `www.samplesite.org` va concatena calea solicitată la directorul său rădăcină (*webroot*). Pe un sistem *Debian/Ubuntu* care rulează un server Apache, directorul rădăcină implicit este `/var/www`. Astfel, resursa solicitată de client va fi intrarea din sistemul local de fişiere `/var/www/pub/file.html`. În continuare, serverul web va citi fişierul şi îl va transmite clientului. Răspunsul va conţine diverse antete necesare şi fişierul efectiv.

Imaginea următoare este o reprezentare grafică a modului de funcţionare a unui server web:



9-1: Funcţionarea serviciului web

9.2 *Apache HTTP Server*

Apache HTTP Server este actualmente cel mai utilizat server de web. Conform sondajelor realizate de către NetCraft, în iunie 2008, 49.12% din serverele web rulau Apache[1]. Dincolo de faptul că este principalul server web, cu o dominaţie şi mai accentuată pe sisteme Linux/UNIX, motive suplimentare pentru studierea Apache sunt şi performanţa ridicată a acestuia, numărul mare de opţiuni de configurare, posibilitatea adăugării de noi caracteristici (majoritatea sub formă de module compilate), o serie de utilitare asociate şi integrarea facilă cu alte aplicaţii.

Versiunea Apache 2.x a fost o rescriere substanţială a codului versiunii Apache 1.x, adăugând multe îmbunătăţiri. Acestea includ folosirea *thread*-urilor UNIX, suport îmbunătăţit

[1] http://news.netcraft.com/archives/web_server_survey.html

Material cu drept de autor

pentru platforme non-UNIX (precum Windows), un nou API, suport IPv6, introducerea unui nivel de portabilitate, *Apache Portable Runtime*[1]. Versiunea stabilă curentă a serverului este 2.2.9.

Setul de interfețe de programare (API) pe care Apache îl pune la dispoziție este cel care asigură extensibilitatea acestuia prin intermediul modulelor. Pachetul principal conține serverul HTTP, urmând ca solicitările suplimentare să fie satisfăcute prin adăugarea de noi module, precum `mod_ssl`, `mod_perl`, `mod_php`, `mod_auth`, etc.

Versiunea de Apache folosită pe parcursul acestui capitol este 2.2, disponibilă în versiunea stabilă Debian Etch, în Debian Lenny şi ultimele versiuni de Ubuntu.

9.2.1 Instalare

Instalarea versiunii Apache 2.x se realizează în mod obişnuit folosind `apt-get`:

```
ragnarok:~# apt-get install apache2
Reading package lists... Done
Building dependency tree... Done
The following extra packages will be installed:
  apache2-mpm-worker apache2-utils apache2.2-common
The following NEW packages will be installed:
  apache2 apache2-mpm-worker apache2-utils apache2.2-common
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 1765kB of archives.
After unpacking 4547kB of additional disk space will be used.
Do you want to continue [Y/n]? Y
[...]
Module authz_groupfile installed; run /etc/init.d/apache2 force-reload to enable.
Module authn_file installed; run /etc/init.d/apache2 force-reload to enable.
Module authz_host installed; run /etc/init.d/apache2 force-reload to enable.

Setting up apache2-mpm-worker (2.2.3-4+etch1) ...
Starting web server (apache2)....

Setting up apache2 (2.2.3-4+etch1) ...
```

Exceptând componentele de bază, se instalează şi o serie de module şi se porneşte serverul. După cum se observă, se instalează 4 pachete. Acestea sunt:

- `apache2`: conține interfața control a serverului web;
- `apache2-mpm-worker`: conține implementarea de tip *threading worker* a serverului;
- `apache2-util`: conține o serie de utilitare folositoare unui server web (`logresolve`, `htpasswd`, `rotatelogs`, etc.);
- `apache2.2-common`: conține modulele Apache2 standard, incluzând suportul SSL.

9.2.2 Interacțiunea cu serverul web

Există mai multe posibilități de a interacționa cu serverul:

- comanda `apache2` este principala interfață de lucru cu serverul Apache; aceasta permite pornirea, oprirea şi repornirea serverului cu posibilitatea precizării unor opțiuni de configurare dinamică şi a unor informații despre server (lista de module compilate, lista de directive):

```
ragnarok:~# /usr/sbin/apache2
Usage: /usr/sbin/apache2 [-D name] [-d directory] [-f file]
                         [-C "directive"] [-c "directive"]
                         [-k start|restart|graceful|graceful-stop|stop]
                         [-v] [-V] [-h] [-l] [-L] [-t] [-S]
[...]
ragnarok:~# apache2 -l
Compiled in modules:
  core.c
  mod_log_config.c
  mod_logio.c
  worker.c
  http_core.c
  mod_so.c
ragnarok:~# apache2 -k stop
ragnarok:~# apache2 -k start
ragnarok:~# apache2 -k restart
```

[1] http://apr.apache.org/

Material cu drept de autor

- comanda `apache2ctl` este o interfaţă de control a serverului Apache; este comanda preferată pentru pornirea şi repornirea serverului, permiţând şi verificarea corectitudinii fişierului de configurare:

```
ragnarok:/home/razvan# apache2ctl stop
ragnarok:/home/razvan# apache2ctl start
ragnarok:/home/razvan# apache2ctl restart
ragnarok:/home/razvan# apache2ctl configtest
Syntax OK
```

- `/etc/init.d/apache2` este un script care interfaţează interacţiunea cu serverul; poate fi folosit pentru pornirea, repornirea, oprirea serverului, şi pentru recitirea fişierului de configurare:

```
ragnarok:/home/razvan# /etc/init.d/apache2
Usage: /etc/init.d/apache2 {start|stop|restart|reload|force-reload}
ragnarok:/home/razvan# /etc/init.d/apache2 stop
Stopping web server (apache2)....
ragnarok:/home/razvan# /etc/init.d/apache2 start
Starting web server (apache2)....
ragnarok:/home/razvan# /etc/init.d/apache2 restart
Forcing reload of web server (apache2)... waiting .
ragnarok:/home/razvan#
```

Fişierele de jurnalizare pentru server sunt `/var/log/apache/error.log` şi `/var/log/apache/access.log`.

Fişierele de configurare se găsesc în directorul `/etc/apache2/`. Structura acestor fişiere va fi prezentată în continuare.

9.2.3 Configurare globală

După cum s-a precizat, configurarea serverului Apache2 se realizează prin intermediul fişierelor din `/etc/apache2/`. Acest director conţine mai multe intrări după cum se poate vedea în listarea de mai jos. Dacă la versiunea Apache1.x exista un singur fişier de configurare, nevoia de modularitate a condus la apariţia mai multor intrări cu roluri bine stabilite. Fişierele de configurare conţin directive simple[1], precum:

```
User www-data
```

sau directive compuse, denumite şi secţiuni de configurare[2], precum:

```
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>
```

Rolul fiecărei intrări din directorul `/etc/apache2/` este:

- `apache2.conf` conţine configurările globale pentru serverul Apache2; directivele din acest fişier se referă la funcţionarea serverului (numărul de procese deschise, intervale de timeout etc.) şi la includerea de module care vor afecta toate domeniile administrate de server (*virtual hosts*); `apache2.conf`, fiind fişierul principal de configurare, include toate celelalte fişiere:

```
[…]
# Include module configuration:
Include /etc/apache2/mods-enabled/*.load
Include /etc/apache2/mods-enabled/*.conf

# Include all the user configurations:
Include /etc/apache2/httpd.conf

# Include ports listing
Include /etc/apache2/ports.conf

# Include generic snippets of statements
Include /etc/apache2/conf.d/
[…]
```

1 http://httpd.apache.org/docs/2.2/mod/quickreference.html
2 http://httpd.apache.org/docs/2.2/sections.html

Material cu drept de autor

- `httpd.conf` conţine diferite configurări pentru utilizatori;
- `ports.conf` defineşte portul/porturile pe care serverul ascultă conexiuni;
- `envvars` defineşte variabilele de mediu folosite de `apachectl`;
- `conf.d/` conţine fişiere de configurare pentru diverse servicii care folosesc Apache2; exemple sunt servicii de *webmail*, navigare în *repositories*, *wiki* etc.;
- `mods-available/` defineşte modulele Apache instalate în sistem; fişierele de aici au extensia `.load`, indicând modulul care trebuie încărcat; există şi fişiere `.conf` care definesc directive suplimentare de configurare pentru modul;
- `mods-enabled/` defineşte modulele active pe durata rulării serverului; fişierele de aici sunt legături simbolice către fişierele din `mods-available/`;
- activarea unui modul se face, aşadar, prin crearea unei legături simbolice spre fişierul asociat din `mods-available/` şi repornirea serverului; se poate folosi utilitarul `a2enmod`;
- `sites-available/` defineşte domeniile pentru care serverul poate fi configurat să răspundă la cereri; este folosit pentru *virtual hosting*; în primă fază serverul defineşte un singur domeniu (cel implicit) în cadrul fişierului cu numele `default`;
- `sites-enabled/` defineşte domeniile (*virtual hosts*) pentru care serverul răspunde la cereri; ca şi în cazul modulelor, fişierul asociat unui domeniu este o legătură simbolică spre fişierul din `sites-available/`; adăugarea unui nou domeniu se face prin crearea instanţei asociate în `sites-available/`; activarea acelui domeniu se face prin crearea unei legături simbolice în `sites-enabled/` şi repornirea serverului; se recomandă folosirea utilitarului `a2ensite`.

ATENŢIE: *Fişierele specificate sunt specifice distribuţiilor Debian/Ubuntu. Deşi alte distribuţii au, de asemenea, o structură a fişierelor de configurare, locaţia şi denumirea acestora pot diferi.*

9.2.3.1 Configurarea domeniului implicit

După cum s-a precizat, domeniile gestionate de Apache sunt servite prin intermediul *virtual hosts*. În primă fază Apache gestionează un domeniu implicit definit în fişierul `/etc/apache/sites-available/default`. Activarea acestui domeniu este implicită prin existenţa legăturii simbolice `/etc/apache/sites-enabled/000-default`.

Conţinutul implicit al fişierului `/etc/apache/sites-available/default` este prezentat în continuare:

```
ragnarok:~# cat /etc/apache2/sites-enabled/000-default
NameVirtualHost *
<VirtualHost *>
        ServerAdmin webmaster@localhost

        DocumentRoot /var/www/
        <Directory />
                Options FollowSymLinks
                AllowOverride None
        </Directory>
        <Directory /var/www/>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order allow,deny
                allow from all
                # This directive allows us to have apache2's default start page
                # in /apache2-default/, but still have / go to the right place
                RedirectMatch ^/$ /apache2-default/
        </Directory>
        [...]
        ErrorLog /var/log/apache2/error.log

        [...]
</VirtualHost>
```

Nu se va insista asupra directivelor legate de *virtual hosting* întrucât vor fi prezentate în secţiunea TODO.

O directivă importantă este `DocumentRoot`. Aceasta precizează directorul rădăcină (*webroot*) de unde vor fi recuperate resursele cerute de clienţi. Directorul rădăcină pentru

Material cu drept de autor

domeniul implicit este `/var/www`. Acest lucru înseamnă că o cerere de forma `GET /cale/catre/resursa HTTP/1.0` va impune serverului transmiterea `fișierului /var/www/cale/catre/resursa către client.`

Directiva `Directory` specifică proprietăți pentru un anumit director. Pentru directorul `/var/www`, directiva specifică proprietățile de mai jos:

```
        Options Indexes FollowSymLinks MultiViews
```

Directiva `Options` impune folosirea fișierelor index din director [vezi 9.2.3.2], urmărirea legăturilor simbolice existente în director și adăugarea unei extensii la cale.

```
        AllowOverride None
```

Directiva `AllowOverride` este folosită în tandem cu fișierul `.htaccess` care configurează drepturi de acces în Apache [vezi 9.2.3.5]. Aceasta precizează care directive definite în fișierul `.htaccess` pot suprascrie alte directive. În cazul de față, folosirea `None` face inutilă prezența unui fișier `.htaccess`.

```
        Order allow,deny
        allow from all
```

Directivele `allow` și `deny` sunt folosite pentru a permite sau refuza accesul la director. Directiva `Order` indică ordinea în care acestea vor fi evaluate. Dacă ar exista o directivă `deny from all`, atunci orice acces ar fi refuzat pentru că directiva `deny` este analizată ultima.

```
        RedirectMatch ^/$ /apache2-default/
```

Directiva `RedirectMatch` realizează redirectarea pentru o cale către resursă dată. În cazul de față se realizează redirectarea pentru resursa `/`. Acest lucru înseamnă ca o cerere de forma `GET / HTTP/1.0` va fi echivalentă cu cererea `GET /apache2-default/ HTTP/1.0`.

După cum se observă, căile către resurse din cererile HTTP pot fi date sub formă de directoare. În această situație serverul web are două variante de răspuns:

- transmiterea conținutului directorului;
- transmiterea unei resurse implicite, în cazul în care aceasta se află în director.

A doua variantă are prioritate în fața primei. Resursa implicită este, de obicei, un fișier cu numele `index.html` sau altul precizat prin directiva `DirectoryIndex` din modulul `mod_dir`. Mai multe detalii sunt precizate în secțiunea 9.2.3.2.

Un lucru de reținut este faptul că directivele prezente în cadrul directivei compuse `<VirtualHost> ... </VirtualHost>` pot suprascrie directivele din fișierul global de configurare `/etc/apache/apache2.conf`. În cazul de față, s-a specificat fișierul de jurnalizare a erorilor pentru domeniul gestionat implicit:

```
        ErrorLog /var/log/apache2/error.log
```

9.2.3.2 Configurare module

Facilitățile de bază ale serverului Apache pot fi extinse prin intermediul modulelor. Modulele de bază sunt instalate prin intermediul pachetului `apache2-common`. Fiecare modul este de fapt o bibliotecă partajată; fișierul asociat are extensia `.so`. Pentru adăugarea unui modul se folosește directiva `LoadModule`.

Un modul are asociat un fișier `.load` în care se folosește directiva `LoadModule`. Acest fișier se găsește în directorul `/etc/apache2/mods-available/`. De exemplu, fișierul asociat modulului SSL este `/etc/apache2/mods-available/ssl.load`:

```
        ragnarok:~# cat /etc/apache2/mods-available/ssl.load
        LoadModule ssl_module /usr/lib/apache2/modules/mod_ssl.so
```

Material cu drept de autor

Biblioteca partajată asociată este `/usr/lib/apache2/modules/mod_ssl.so`. De obicei, numele bibliotecii este `mod_nume.so`, unde `nume` este numele modulului.

Unele module, precum SSL, au asociate un fişier de configurare cu extensia `.conf`. Un astfel de fişier de configurare conţine directive specifice modulului respectiv:

```
ragnarok:~# cat /etc/apache2/mods-available/ssl.conf
<IfModule mod_ssl.c>
[...]
SSLRandomSeed startup builtin
SSLRandomSeed startup file:/dev/urandom 512
SSLRandomSeed connect builtin
SSLRandomSeed connect file:/dev/urandom 512
[...]
</IfModule>
```

Pentru ca un modul să fie utilizat de serverul web se creează o legătură simbolică în `/etc/apache2/mods-enabled/` către fişierul `.load` şi, eventual, `.conf` din directorul `mods-available/`. Se recomandă folosirea utilitarului `a2enmod`:

```
ragnarok:~# a2enmod ssl
Module ssl installed; run /etc/init.d/apache2 force-reload to enable.
```

Configurarea unui modul, realizată fie în fişiere `.conf` specializate sau fişiere globale, foloseşte directive încadrate de directiva `IfModule`. Directiva `IfModule` testează dacă modulul respectiv este încărcat în server. Astfel, pentru a configura `mod_ssl` se foloseşte o construcţie de forma:

```
<IfModule mod_userdir.c>
...
</IfModule>
```

iar pentru a configura `mod_alias` se foloseşte o construcţie de forma:

```
<IfModule mod_alias.c>
...
</IfModule>
```

În continuare sunt prezentate câteva aspecte ale configurării unor module importante Apache: `mod_dir`, `mod_alias`, `mod_userdir`, `mod_cgi`.

9.2.3.2.1 Configurare fişiere index

Fişierele index sunt fişierele care sunt afişate în cazul în care calea dintr-o cerere HTTP este un director. Astfel, dacă un utilizator foloseşte URL-ul `http://www.example.com/pub/`, se transmite cererea `GET /pub/ HTTP/1.0` sau `GET /pub/ HTTP/1.1` către server. Serverul va trebui să ofere resursa `/var/www/pub/` care este un director. În acest moment, serverul caută în director un fişier index şi îl transmite clientului.

Configurarea fişierelor index se realizează prin intermediul modulului `mod_dir`. Modulul este activat implicit:

```
ragnarok:~# ls /etc/apache2/mods-enabled/dir*
/etc/apache2/mods-enabled/dir.conf  /etc/apache2/mods-enabled/dir.load
```

Fişierul `/etc/apache2/mods-available/dir.conf` către care punctează legătura simbolică `/etc/apache2/mods-enabled/dir.conf` permite configurarea fişierelor index:

```
ragnarok:~# cat /etc/apache2/mods-available/dir.conf
<IfModule mod_dir.c>
        DirectoryIndex index.html index.cgi index.pl index.php index.xhtml
</IfModule>
```

Configurarea existentă impune serverului căutarea pe rând a fişierelor `index.html`, `index.cgi`, `index.pl`, `index.php`, `index.xhtml` în cazul în care calea către resursă este un director. Dacă se doreşte adăugarea ca fişier index un fişier cu numele `index.txt` va trebui alterată configuraţia existentă:

Material cu drept de autor

```
ragnarok:~# cat /etc/apache2/mods-available/dir.conf
<IfModule mod_dir.c>
        DirectoryIndex index.html index.cgi index.pl index.php index.xhtml
</IfModule>
```

9.2.3.2.2 Configurare aliasuri

De multe ori este dificil, incomod, sau chiar imposibil să se stocheze unele resurse servite de serverul web în ierarhia dată de `/var/www`. Se poate dori, spre exemplu, ca accesul la documentaţia existentă în `/usr/share/doc` să se facă prin intermediul unui URL de forma `http://localhost/doc/`. Soluţia la această problemă este folosirea aliasuri.

Un alias asociază o nume de resursă ce poate fi parte a unui URL cu o resursă din sistemul de fişiere. Spre exemplu, dacă se doreşte ca folosirea URL-ului `http://localhost/doc/` să conducă la afişarea resurselor din `/usr/share/doc`, se va utiliza o directivă de forma:

```
Alias /doc/ /usr/share/doc
```

Folosirea de aliasuri este condiţionată de existenţa modulului `mod_alias`. Acest modul este încărcat implicit la pornirea serverului Apache2:

```
ragnarok:~# ls /etc/apache2/mods-enabled/alias.*
/etc/apache2/mods-enabled/alias.load
```

Configurarea de aliasuri se face în fişierul global de configurare `/etc/apache2/apache2.conf`. Directivele de configurare sunt încadrate de directiva `<IfModule>`. De obicei configurările se realizează la nivel de domeniu virtual deoarece directorul `webroot` diferă. Configuraţia de alias pentru domeniul implicit este prezentată în continuare:

```
ragnarok:~# cat /etc/apache2/sites-enabled/000-default
NameVirtualHost *
<VirtualHost *>
    [...]
    Alias /doc/ "/usr/share/doc/"
    <Directory "/usr/share/doc/">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
        Allow from 127.0.0.0/255.0.0.0 ::1/128
    </Directory>
</VirtualHost>
```

Se observă crearea aliasului `/doc/` aşa cum s-a precizat mai sus. O directivă `Alias` are asociată o directivă `Directory` în care se precizează proprietăţile directorului peste care s-a realizat aliasul.

9.2.3.2.3 Configurare UserDir

De obicei, un utilizator al sistemului de operare doreşte publicarea unor pagini web fără a fi nevoie de a contacta administratorul sistemului. Utilizatorul creează un director local care va fi automat folosit de serverul web.

Implementarea unei astfel de facilitaţi se realizează în Apache prin intermediul modulului UserDir: `mod_userdir`. Acest modul nu este activat implicit în cadrul Apache2 şi trebuie activat manual:

```
ragnarok:~# cd /etc/apache2/mods-enabled/
ragnarok:/etc/apache2/mods-enabled# ln -s ../mods-available/userdir.conf userdir.conf
ragnarok:/etc/apache2/mods-enabled# ln -s ../mods-available/userdir.load userdir.load
ragnarok:/etc/apache2/mods-enabled# apache2ctl restart
```

Crearea manuală a legăturilor simbolice este destul de dificilă şi susceptibilă la erori. Soluţia recomandată este folosirea scripturilor `a2enmod` şi `a2dismod` oferite de Debian pentru activarea, respectiv dezactivarea modulelor din server:

```
ragnarok:~# a2dismod userdir
```

Material cu drept de autor

```
Module userdir disabled; run /etc/init.d/apache2 force-reload to fully disable.
ragnarok:~# a2enmod userdir
Module userdir installed; run /etc/init.d/apache2 force-reload to enable.
```

Serverul trebuie repornit pentru încărcarea modulului.

Configurația pentru modulul `UserDir` se află în `/etc/apache2/mods-available/userdir.conf`:

```
ragnarok:~# cat /etc/apache2/mods-available/userdir.conf
<IfModule mod_userdir.c>
        UserDir public_html
        UserDir disabled root

        <Directory /home/*/public_html>
                AllowOverride FileInfo AuthConfig Limit
                Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
        </Directory>
</IfModule>
```

Ca de obicei, directivele de configurare sunt încadrate de directiva `IfModule`.

Directiva

```
UserDir public_html
```

precizează ca *webroot* pentru fiecare utilizator directorul local `public_html`. Posibilitatea publicării de documente dintr-un director propriu de utilizatorul privilegiat este dezactivată:

Opțiunile pentru director sunt date de directiva `Directory`:

```
<Directory /home/*/public_html>
```

Câmpul * este înlocuit cu numele utilizatorului.

Astfel, directorul `webroot` local al utilizatorului `andrei` folosit pentru publicarea de resurse prin intermediul serverului web este `/home/andrei/public_html`.

Accesarea directorului webroot local se realizează prin intermediul unui URL de forma `http://www.example.com/~andrei/`. Astfel, accesul la pagina `/home/andrei/public_html/pub/file.html` se realizează prin intermediul URL-ului `http://localhost/~andrei/pub/file.html`.

Ca un exercițiu, se presupune că există un set de utilizatori care au directorul `home` în `/home/students/username`, unde `username` este numele utilizatorilor. Administratorul de sistem decide că și utilizatori vor putea publica documente web. În această situație configurația modulului `UserDir` va fi:

```
<IfModule mod_userdir.c>
        [...]
        <Directory /home/students/*/public_html>
                AllowOverride FileInfo AuthConfig Limit
                Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
        </Directory>
</IfModule>
```

Configurația a fost adăugată la configurația anterioară. Serverul web trebuie repornit pentru a încărca modificările efectuate.

Se observă că s-au stabilit proprietățile pe directorul `/home/students/*/public_html`, unde * va fi înlocuit cu numele utilizatorului. Astfel, dacă utilizatorul `diana` are directorul home în `/home/students/diana`, folosirea URL-ului `http://localhost/~diana/docs/chapter.doc` va conduce la obținerea fișierului `/home/students/diana/public_html/docs/chapter.doc`.

9.2.3.2.4 Configurare CGI

CGI (*Common Gateway Interface*) este un protocol care permite interfațarea între o aplicație și un server web. Acest lucru permite transmiterea cererilor de la clienți către aplicație. După rularea aplicației, serverul web va întoarce rezultatul acesteia către client.

Material cu drept de autor

Un program CGI este specificat prin intermediul unui URL, spre exemplu `http://www.example.com/simple.cgi`. Folosirea acestui URL impune serverului web rularea programului asociat. Serverul web colectează rezultatul rulării programului şi îl transmite clientului. Acest lucru are dezavantajul rulării unui proces separat pentru fiecare instanţă de cerere. O soluţie este folosirea de module precum `mod_php` sau `mod_perl` [vezi 9.2.3.3] care permit integrarea unui interpretor în serverul web. O altă soluţie este folosirea de limbaje de programare precum C care pot atinge o eficienţă mai mare prin terminarea mai rapidă a execuţiei.

Suportul de CGI pentru Apache este obţinut prin intermediul modulului `mod_cgi`. Întrucât nu este activat implicit, trebuie activat manual:

```
ragnarok:~# a2enmod cgi
Module cgi installed; run /etc/init.d/apache2 force-reload to enable.
ragnarok:/etc/apache2/mods-enabled# apache2ctl restart
```

Configurarea directorului ce conţine scripturile CGI se realizează la nivel de domeniu virtual. Astfel, configuraţia CGI pentru domeniul virtual implicit este:

```
ragnarok:~# cat /etc/apache2/sites-enabled/000-default
NameVirtualHost *
<VirtualHost *>
        [...]
        ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
        <Directory "/usr/lib/cgi-bin">
                AllowOverride None
                Options ExecCGI -MultiViews +SymLinksIfOwnerMatch
                Order allow,deny
                Allow from all
        </Directory>
        [...]
</VirtualHost>
```

Directorul de unde se vor executa scripturile CGI este definit prin intermediul directivei `ScriptAlias` şi este, implicit, `/usr/lib/cgi-bin/`. Accesul la scriptul `/usr/lib/cgi-bin/sample.cgi` se realizează prin intermediul unui URL de forma `http://localhost/cgi-bin/script.cgi`. Opţiunea `ExecCGI` este cea care specifică serverului execuţia scriptului prin intermediul unei aplicaţii externe.

Un script simplu CGI este următorul program C:

```
ragnarok:~# cat sample.c
#include <stdio.h>

int main (void)
{
        printf ("Content-type: text/html\n\n");
        printf (
                        "<html>\n"
                        "\t<head>\n"
                        "\t\t<title>Pagina mea</title>\n"
                        "\t</head>\n"
                        "\t<body>\n"
                        "\t\t<h1>Hello, World!</h1>\n"
                        "\t</body>\n"
                        "</html>\n"
            );

        return 0;
}
ragnarok:~# gcc -Wall sample.c -o sample.cgi
ragnarok:~# mkdir /usr/lib/cgi-bin/
ragnarok:~# cp sample.cgi /usr/lib/cgi-bin/
```

Programul C a fost compilat, numele executabilului fiind `sample.cgi`; extensia `.cgi` este opţională. Executabilul a fost copiat în `/usr/lib/cgi-bin`. Întrucât directorul nu exista anterior a fost creat.

Pentru testare putem folosi `telnet` sau `netcat`:

```
ragnarok:~# telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^]'.
```

Material cu drept de autor

```
GET /cgi-bin/sample.cgi HTTP/1.0

HTTP/1.1 200 OK
Date: Sun, 23 Sep 2007 14:02:19 GMT
Server: Apache/2.2.3 (Debian)
Content-Length: 102
Connection: close
Content-Type: text/html; charset=UTF-8

<html>
        <head>
                <title>Pagina mea</title>
        </head>
        <body>
                <h1>Hello, World!</h1>
        </body>
</html>
Connection closed by foreign host.
```

Cererea trimisă serverului a fost `GET /cgi-bin/sample.cgi HTTP/1.0`. Serverul trimite ca răspuns clientului mesajul `HTTP/1.1 200 OK` împreună cu un antet de mesaj şi pagina web rezultată prin rularea scriptului CGI.

9.2.3.3 Instalare module

Alte module pot fi instalate cu ajutorul utilitarului apt-`get`. Pachetele asociate modulelor Apache sunt denumite `libapache2-mod-nume`, unde `nume` identifică modulul. O listă a modulelor instalabile Apache poate fi obținută cu ajutorul comenzii `apt-cache`:

```
ragnarok:/etc/apache2# apt-cache search libapache2-mod-
libapache2-mod-auth-kerb - apache2 module for Kerberos authentication
libapache2-mod-auth-pam - module for Apache2 which authenticate using PAM
[...]
libapache2-mod-bt - BitTorrent tracker for the Apache2 web server
libapache2-mod-chroot - run Apache in a secure chroot environment
libapache2-mod-musicindex - Browse, stream, download and search through MP3/Ogg/FLAC
files
[...]
libapache2-mod-perl2 - Integration of perl with the Apache2 web server
libapache2-mod-python - Apache 2 module that embeds Python within the server
libapache2-mod-ruby - Embedding Ruby in the Apache2 web server
libapache2-mod-php4 - server-side, HTML-embedded scripting language (apache 2 module)
libapache2-mod-php5 - server-side, HTML-embedded scripting language (apache 2 module)
```

Pentru exemplificare se vor instala modulele `mod_php5` şi `mod_perl2` care permit rularea de scripturi PHP sau Perl prin intermediul serverului web.

9.2.3.3.1 mod_php

Suportul PHP se obține prin intermediul modulului `mod_php5`:

```
ragnarok:~# apt-get install libapache2-mod-php5
Reading package lists... Done
Building dependency tree... Done
The following extra packages will be installed:
  apache2-mpm-prefork php5-common
Suggested packages:
  php-pear
The following packages will be REMOVED:
  apache2-mpm-worker
The following NEW packages will be installed:
  apache2-mpm-prefork libapache2-mod-php5 php5-common
0 upgraded, 3 newly installed, 1 to remove and 0 not upgraded.
Need to get 3044kB of archives.
After unpacking 5870kB of additional disk space will be used.
Do you want to continue [Y/n]? Y
[...]
```

Se observă că se înlocuieşte modulul de *threading worker* cu modulul *prefork*. Modulul este automat activat după instalare:

```
ragnarok:~# ls /etc/apache2/mods-enabled/php5.*
/etc/apache2/mods-enabled/php5.conf  /etc/apache2/mods-enabled/php5.load
```

Fişierul de configurare `php5.conf` defineşte extensiile de fişiere pentru care va fi utilizat modulul:

Material cu drept de autor

```
ragnarok:~# cat /etc/apache2/mods-enabled/php5.conf
<IfModule mod_php5.c>
  AddType application/x-httpd-php .php .phtml .php3
  AddType application/x-httpd-php-source .phps
</IfModule>
```

Pentru testare folosim fişierul `test.php` de mai jos:

```
ragnarok:~# cat test.php
<?php
        phpinfo ();
?>
ragnarok:~# cp test.php /var/www/
```

Folosirea URL-ului `http://localhost/test.php` duce la afişarea unei pagini de informaţii despre PHP.

9.2.3.3.2 mod_perl

Suportul *Perl* se obţine prin instalarea modulului `mod_perl2`:

```
ragnarok:~# apt-get install libapache2-mod-perl2
```

La fel ca modulul `mod_php5`, `mod_perl2` este automat adăugat la lista de module încărcate. Pentru încărcarea modulului trebuie repornit serverul.

```
ragnarok:~# apache2ctl restart
```

Un fişier de test este următorul:

```
ragnarok:~# cat test.pl
#!/usr/bin/perl

print "Content-type: text/plain\r\n\r\n";
print "Hello, world!\n";
```

Scripturile *Perl* sunt folosite ca scripturi CGI. Pentru aceasta, scriptului *Perl* `test.pl` îi sunt acordate drepturi de execuţie, după care este copiat în `/usr/lib/cgi-bin/`:

```
ragnarok:~# chmod a+x test.pl
ragnarok:~# cp test.pl /usr/lib/cgi-bin/
```

9.2.3.4 Configurare drepturi în sistemul de fişiere

Două directive din fişierul de configurare precizează utilizatorul şi grupul din sistemul de operare folosite de serverul Apache pentru accesarea resurselor web: `User` şi `Group` definite în fişierul principal de configurare:

```
User www-data
Group www-data
```

Dacă utilizatorul `www-data` nu are drepturi suficiente pe resursa cerută atunci nu serverul îi va returna un mesaj de tip `Forbidden`. De obicei, un fişier va trebui să ofere drepturi de acces de citire utilizatorului `www-data` pentru ca acesta să poată fi transmis clientului. Dreptul de scriere este necesar pentru upload de fişiere şi este folosit mai rar; un exemplu de utilizare a dreptului de scriere este un *wiki*.

Înainte de a prezenta modul în care serverul web interacţionează cu drepturile pe sistemul de fişiere, vor fi reamintite mecanismele de drepturi de acces pe un sistem Unix.

9.2.3.4.1 Recapitulare drepturi Unix

După cum se ştie, un sistem Unix are trei tipuri de drepturi:

- **citire** (*read* – r): un fişier poate fi vizualizat şi se poate lista conţinutul unui director;
- **scriere** (*write* – w): un fişier poate fi editat şi se pot crea noi intrări într-un director; intrările în sistemul de fişiere pot fi şterse;

Material cu drept de autor

- **execuţie** (*execute* – x): un fişier poate fi executat şi un director poate fi parte a unei căi; informal spus, se poate „trece" printr-un director.

Orice utilizator poate avea oricare din aceste drepturi de acces asupra unei intrări în sistemul de fişiere. Pentru o gestiune mai facilă, un fişier are grupat utilizatorii în trei categorii:

- **user** (u): utilizatorul care deţine fişierul
- **group** (g): grupul care deţine fişierul
- **other** (o): ceilalţi utilizatori

Schimbarea utilizatorului sau grupului care deţine fişierul se realizează cu ajutorul comenzii `chown`.

Fiecare din aceste trei categorii au asociate drepturi de citire, scriere şi execuţie. De obicei drepturile pe un fişier sunt date prin concatenarea drepturilor pentru entităţile de mai sus. Astfel gruparea de drepturi `rwx r-x -wx` înseamnă că utilizatorul are toate drepturile asupra fişierului, grupul are drepturi de citire şi execuţie, iar ceilalţi au drept de scriere şi execuţie.

Drepturile de acces pe o intrare din sistemul de fişiere se vizualizează cu ajutorul comenzii `ls -l`:

```
ragnarok:~# ls -l
total 49
drwx------ 24 root root  792 Feb 24  2007 dir
drwxr-xr-x  2 root root  152 May 31 11:55 Desktop
-rw-r--r--  1 root root  179 Nov 14  2006 dbootstrap_settings
----------  1 root root    6 Jan 21  2007 huhu
-rw-r--r--  1 root root 1336 Nov 14  2006 install-report.template
-rw-------  1 root root 2006 Nov 14  2006 mbox
-rwxr-xr-x  1 root root 7158 Sep 23 17:01 sample.cgi
```

În *listing* se observă că utilizatorul care deţine intrările este `root`, iar grupul deţinător este tot `root`. Utilizatorul `root` are toate drepturile pe fişierul `dir`, iar grupul şi ceilalţi utilizatori nu au niciun drept. Fişierul `sample.cgi` poate fi executat de către orice utilizator, având drept de execuţie pentru utilizator, grup şi ceilalţi.

Înşiruirea de drepturi asupra unei intrări în sistemul de fişiere poate fi folosită în format binar, asociindu-se 9 biţi: câte un bit pentru prezenţa sau nu a unui drept. Astfel, formatul literal `rwx r-x -wx` are asociat formatul binar `111 101 011`. Se preferă formatul octal al drepturilor, care pentru exemplul anterior este `753`.

Drepturile pe o intrare în sistemul de fişiere pot fi alterate folosind comanda `chmod`. În cele ce urmează se vor prezenta mai multe exemple de rulare a chmod pentru schimbarea succesivă a drepturilor de acces pe fişiere:

- toate drepturile pentru utilizator, niciun drept pentru ceilalţi

```
ragnarok:~# ls -l test.txt
-rw-r--r-- 1 root root 0 Sep 23 19:23 test.txt
ragnarok:~# chmod 700 test.txt
ragnarok:~# ls -l test.txt
-rwx------ 1 root root 0 Sep 23 19:23 test.txt
```

- drept de citire şi scriere pentru utilizator, drept de citire pentru grup, drept de scriere pentru ceilalţi

```
ragnarok:~# chmod 642 test.txt
ragnarok:~# ls -l test.txt
-rw-r---w- 1 root root 0 Sep 23 19:23 test.txt
```

- drept de citire şi execuţie pentru utilizator, drept de citire şi scriere pentru grup, drept de scriere şi execuţie pentru ceilalţi

```
ragnarok:~# chmod 563 test.txt
ragnarok:~# ls -l test.txt
-r-xrw--wx 1 root root 0 Sep 23 19:23 test.txt
```

- revenirea la drepturile iniţiale: drept de citire şi scriere pentru utilizator, drept de citire pentru grup şi pentru ceilalţi

```
ragnarok:~# chmod 644 test.txt
```

Material cu drept de autor

```
ragnarok:~# ls -l test.txt
-rw-r--r-- 1 root root 0 Sep 23 19:23 test.txt
```

9.2.3.4.2 Configurare drepturi server web

După cum s-a precizat, pentru ca o resursă să poată fi accesată de serverul web, trebuie ca utilizatorul sau grupul `www-data` să aibă drept de citire asupra acesteia.

Fie următoarea situație:

```
ragnarok:/var/www# ls -l
total 0
-rw-r--r-- 1 root root   0 Sep 23 19:32 brad.txt
-rw-r----- 1 root root   0 Sep 23 19:32 fag.txt
```

Fișierul `brad.txt` va putea fi accesat de serverul web, în timp ce fișierul `fag.txt`. Utilizatorul `www-data` face partea din categoria de utilizatori others, adică ultimele 3 drepturi. În cazul `brad.txt` aceste drepturi sunt `r--` iar în cazul `fag.txt` sunt `---` (niciun drept).

Dacă se schimbă grupul pentru `fag.txt` în `www-data`, atunci și acesta va putea fi accesat de serverul web:

```
ragnarok:/var/www# chown root:www-data fag.txt
ragnarok:/var/www# ls -l
total 0
-rw-r--r-- 1 root root       0 Sep 23 19:32 brad.txt
-rw-r----- 1 root www-data   0 Sep 23 19:32 fag.txt
```

Se va adăuga un director la structura deja existentă:

```
ragnarok:/var/www# mkdir carpen
ragnarok:/var/www# touch carpen/molid.txt
ragnarok:/var/www# touch carpen/frasin.txt
ragnarok:/var/www# ls -ld carpen/
drwxr-xr-x 2 root root 112 Sep 23 19:47 carpen/
ragnarok:/var/www# ls -l carpen/
total 0
-rw-r--r-- 1 root root 0 Sep 23 19:47 frasin.txt
-rw-r--r-- 1 root root 0 Sep 23 19:47 molid.txt
```

Directorul `carpen/` oferă drepturi de citire și execuție pentru ceilalți utilizatori astfel că se poate afișa conținutul său. Drept urmare, folosirea URL-ului `http://localhost/carpen/` va conduce la listarea conținutului acestui director. Fișierele `frasin.txt` și `molid.txt` pot fi accesate prin intermediul paginii afișate. Dacă se schimbă drepturile pe un fișier:

```
ragnarok:/var/www# chmod 640 carpen/molid.txt
```

atunci utilizatorul `www-data` nu va mai avea drept de citire. Fișierul va fi totuși afișat în momentul folosirii URL-ului `http://localhost/carpen/`, dar nu va putea fi accesat. Cu alte cuvinte schimbarea drepturilor de acces pe un fișier nu schimbă comportamentul directorului care îl conține.

Listarea directorului este dezactivată dacă există un fișier index [vezi 9.2.3.2]. Astfel, dacă se creează fișierul `index.html`, se va face implicit afișarea acestuia, nu a conținutului directorului.

```
ragnarok:/var/www# touch carpen/index.html
```

Fișierele `frasin.txt` și `molid.txt` vor putea fi referite, în acest caz, numai dacă li se cunoaște calea completă. Altfel spus, vor putea fi recuperate cu ajutorul URL-urilor `http://localhost/carpen/frasin.txt`, respectiv `http://localhost/carpen/molid.txt`.

De multe ori se dorește dezactivarea listării conținutul unui director fără a fi nevoie de prezența unui fișier `index.html`. Pentru aceasta trebuie eliminat dreptul de citire de director, dar păstrat cel de execuție (de parcurgere a directorului):

```
ragnarok:/var/www# chmod 711 carpen/
```

Material cu drept de autor

După comanda de mai sus, utilizatorul `www-data` va avea doar drept de execuţie asupra directorului `carpen/`. Dacă acest director va conţine un fişier index, acesta va fi afişat. Altfel se va afişa un mesaj de tip `Forbidden`. Fişierele dintr-un astfel de director pot fi accesate doar prin precizarea căii complete în URL.

9.2.3.5 Configurare opţiuni per director. Fişierul .htaccess

Directoarele accesibile prin intermediul serverului web pot avea asociate opţiuni prin intermediul unui fişier existent în acest director. Acest fişier poartă numele de `.htaccess`[1]. Prezenţa acestui fişier într-un director înseamnă parcurgerea acestuia şi interpretarea directivelor existente pentru acel director.

Fişierul `.htaccess` este folosit atunci când un utilizator nu are acces la fişierul principal de configurare. În cazul în fişierul principal de configurare poate fi folosit pentru a configura un director, se recomandă să nu se folosească fişierul `.htaccess`. Opţiunile per director sunt configurate în fişierul principal cu ajutorul directivei `<Directory>`.

Configuraţiile din `.htaccess` sunt aplicate directorului curent şi tuturor subdirectoarelor acestuia. Configuraţiile din fişierele `.htaccess` din subdirectoare vor suprascrie configurările din directorul curent. Directiva `AllowOverride` descrisă în secţiunea 9.2.3.1 este folosită pentru a preciza directivele din fişierul principal de configurare pe care fişierul `.htaccess` le poate suprascrie. Astfel, folosirea

```
AllowOverride None
```

ignoră prezenţa fişierului `.htaccess`, pe când folosirea

```
AllowOverride All
```

acordă prioritate directivelor prezente în fişierul `.htaccess`.

Un exemplu este configurarea unui director pentru a rula scripturi CGI. Pentru aceasta fişierul `.htaccess` va avea conţinutul:

```
ragnarok:~# cat /var/www/test/.htaccess
Options +ExecCGI
SetHandler cgi-script
```

Pentru ca aceste configurări să fie active, va trebui ca directorul `/var/www/test` să permită directivele de configurare `Options` şi `FileInfo`:

```
ragnarok:~# cat /etc/apache2/sites-enabled/000-default
NameVirtualHost *
<VirtualHost *>
        [...]
        <Directory /var/www/test/>
                AllowOverride Options FileInfo
        </Directory>
        [...]
</VirtualHost>
```

După aceasta folosirea URL-ului `http://localhost/test/sample.cgi` va conduce la execuţia scriptului `sample.cgi`.

9.2.3.5.1 Folosire directive de autentificare

Fişierul `.htaccess` poate fi folosit pentru autentificarea utilizatorului în momentul în care accesează directorul asociat[2]. Ca şi alte opţiuni, se recomandă utilizarea directivelor de autentificare în cadrul unei directive `Directory` într-un fişier de configurare. Folosirea acestor directive într-un fişier `.htaccess` se foloseşte în momentul în care nu se poate accesa fişierul de configurare.

[1] http://httpd.apache.org/docs/2.2/howto/htaccess.html
[2] http://httpd.apache.org/docs/2.2/howto/auth.html

Material cu drept de autor

Un exemplu de configurare este următorul:

```
ragnarok:~# mkdir /var/www/need_auth
ragnarok:~# vi /var/www/need_auth/.htaccess
ragnarok:~# cat /var/www/need_auth/.htaccess
AuthType Basic
AuthName "Authentication Needed"
AuthUserFile /var/www/need_auth_passwd
Require user mihai roxana
```

Tipul de autentificare folosit este `Basic`. Directiva `AuthName` defineşte domeniul (*Realm*) cu care va fi asociată instanţa curentă de autentificare. Autentificarea foloseşte fişierul precizat prin intermediul directivei `AuthUserFile`, în cazul de faţă `/var/www/need_auth_passwd`. Directiva `Require` specifică modul în care se realizează autentificarea. În cazul de faţă se permite accesul numai utilizatorilor `mihai` şi `roxana`.

Crearea şi completarea fişierului de configurare se realizează cu ajutorul utilitarului `htpasswd`. Astfel, pentru crearea fişierului şi adăugarea utilizatorilor `mihai` şi `roxana` se foloseşte secvenţa de comenzi de mai jos:

```
ragnarok:~# htpasswd -c /var/www/need_auth_passwd mihai
New password:
Re-type new password:
Adding password for user mihai
ragnarok:~# htpasswd /var/www/need_auth_passwd roxana
New password:
Re-type new password:
Adding password for user roxana
```

La prima rulare s-a folosit opţiunea `-c` pentru a crea fişierul de parole. Parolele sunt criptate în fişierul de autentificare:

```
ragnarok:~# cat /var/www/need_auth_passwd
mihai:nzXO9vzU9BcmA
roxana:uJOLmQIETpCZw
```

Pentru a putea folosi directivele de autentificare din fişierul `.htaccess` va trebui folosită directiva `AllowOverride` cu opţiunea `AuthConfig`:

```
<Directory /var/www/need_auth/>
        AllowOverride AuthConfig
</Directory>
```

Folosirea URL-ului `http://localhost/need_auth/` va duce la apariţia unui prompt de autentificare a utilizatorului. Utilizatorul se va putea autentifica cu numele de utilizator şi parola specificate în fişierul `/var/www/need_auth_passwd`, în cazul de faţă `mihai` şi `roxana` şi parolele asociate. După autentificare va putea accesa resursele din director.

9.2.4 Găzduire virtuală

Una dintre cele mai importante facilităţi în Apache este posibilitatea de **găzduire virtuală** (*virtual hosting*)[1]. Actualmente, aceasta este metoda fundamentală de a rula mai multe servicii web - fiecare cu nume şi URL-uri diferite - dar care reprezintă *site*-uri distincte. *Virtual hosting* este folosit pe scară largă în ISP-uri şi *site*-uri de *hosting* care au nevoie să administreze mai multe *site*-uri, dar nu vor să achiziţioneze un calculator nou pentru fiecare în parte.

9.2.4.1 Tipuri de găzduire virtuală

Există două moduri de a implementa găzduirea virtuală a paginilor web: **găzduire virtuală bazată pe adrese IP** (*IP-based virtual hosting*) şi **găzduire virtuală bazată pe nume** (*name-based virtual hosting*). Un caz particular poate fi considerat **găzduirea virtuală bazată pe port**.

[1] http://httpd.apache.org/docs/2.2/vhosts/

În cadrul găzduirii bazate pe adrese IP, fiecare *virtual host* primeşte o adresă IP. Pe anumite sisteme de operare se pot stabili mai multe adrese IP pe o interfaţă de reţea. Pe alte sisteme va fi nevoie de o interfaţă suplimentară pentru fiecare adresă IP.

Totuşi, adresele IP costă şi sunt din ce în ce mai greu de obţinut, astfel încât *browser*-ele moderne pot folosi *virtual hosting* bazat pe nume. Când un server web primeşte o conexiune nu cunoaşte ce nume de staţie (*hostname*) a fost folosit în URL. Pentru a corecta, noua specificaţie HTTP/1.1 adaugă o facilitate prin intermediul căreia *browser*-ele pot spune serverului ce *hostname* folosesc, prin intermediul antetului `Host:`.

Deşi găzduirea virtuală prezintă avantajul economisirii de adrese IP, trebuie ţinut cont de existenţa unor *browser*-e mai vechi care nu au implementată specificaţia HTTP/1.1. Dacă un astfel de *browser* se conectează la un server web cu găzduire virtuală bazată pe nume, nu va trimite antetul `Host:`, aşa că serverul va trebui să răspundă cu o listă de gazde virtuale posibile.

9.2.4.2 Alegerea numelui de staţie

După stabilirea tipului de găzduire virtuală dorit şi, eventual, a unei adrese IP, trebuie ales un nume de staţie şi actualizat corespunzător serverul DNS asociat.

Se presupune că serverul web ascultă conexiuni pe două adrese IP: `99.88.77.66` şi `88.77.66.55`. În acelaşi timp va folosi găzduire virtuală bazată pe adrese IP şi să asculte folosească domeniile `www1.mydomain.com` şi `www2.mydomain.com`. În acest caz trebuie adăugate două intrări în fişierul de configurare DNS ca cele de mai jos. S-a presupus că serverul de nume configurat este autoritar pe domeniul `mydomain.com`:

```
        www1    A       99.88.77.66
        www2    A       88.77.66.55
```

Dacă se doreşte folosirea găzduiri virtuale bazate pe nume folosind ca adresă IP doar `99.88.77.66` atunci vor exista două intrări de forma

```
        www1    A               99.88.77.66
        www2    CNAME           www1
```

O situaţie particulară, folosită la testare, este găzduirea virtuală bazată pe nume cu acces numai de pe sistemul local. Pentru aceasta este nevoie de adăugarea în `/etc/hosts` a unei linii de forma:

```
        127.0.0.1       www.mydomain.com
```

9.2.4.3 Configurare găzduire virtuală în Apache2

După cum s-a specificat şi în secţiunea 9.2.3, configurarea găzduirii virtuale în Apache2 se realizează, pe Debian/Ubuntu, prin intermediul directoarelor `/etc/apache2/sites-available` şi `/etc/apache2/sites-enabled/`. Directorul `sites-available/` conţine câte un fişier pentru fiecare domeniu virtual potenţial. Directorul `sites-enabled/` conţine legături simbolice către fişierele din `sites-available/` precizând, astfel, domeniile virtuale care sunt activate la rularea serverului.

Iniţial, Apache2 are configurat un domeniu virtual implicit în `/etc/apache2/sites-available/000-default`:

```
        raptor:~# ls -l /etc/apache2/sites-enabled/000-default
        lrwxrwxrwx 1 root root 36 Sep 23 00:06 /etc/apache2/sites-enabled/000-default ->
/etc/apache2/sites-available/default
```

Pentru configurarea unui domeniu virtual se creează un fişier în `sites-available/`, după care se creează o legătură simbolică în `sites-enabled/`. Se recomandă cu numele fişierului să fie chiar numele domeniului pentru care se realizează găzduirea virtuală. Pentru activarea şi

Material cu drept de autor

dezactivarea domeniilor virtuale se recomandă folosirea scripturilor `a2ensite`, respectiv `a2dissite` puse la dispoziţie de pachetul de la Debian.

9.2.4.3.1 Directive pentru găzduire virtuală

Înainte de prezentarea unui exemplu de configurare globală, se vor identifica o parte din directivele de configurare pentru găzduire virtuală folosind domeniul implicit:

```
ragnarok:~# cat /etc/apache2/sites-enabled/000-default
NameVirtualHost *
<VirtualHost *>
        ServerAdmin webmaster@localhost

        DocumentRoot /var/www/
        <Directory />
                Options FollowSymLinks
                AllowOverride None
        </Directory>
        <Directory /var/www/>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order allow,deny
                allow from all
                RedirectMatch ^/$ /apache2-default/
        </Directory>
        [...]
</VirtualHost>
```

Din conţinutul fişierului s-au păstrat doar directivele folosite pentru configurarea găzduirii virtuale. Directivele de configurare pentru găzduirea virtuală sunt:

- `NameVirtualHost` indică folosirea găzduirii virtuale; recomandarea[1] pentru găzduirea virtuală bazată pe nume este crearea unui fişier `/etc/apache2/conf.d/virtual.conf` care să conţină opţiunea `NameVirtualHost` urmată de adresa IP folosită pentru găzduire virtuală; caracterul `*` înseamnă că serverul ascultă pe toate interfeţele;
- `<VirtualHost *>` defineşte un domeniu virtual; în cazul în care se foloseşte găzduire virtuală se foloseşte acelaşi parametru ca şi la `NameVirtualHost`, în cazul de faţă `*`;
- `ServerAdmin` defineşte adresa de e-mail a administratorului domeniului;
- `DocumentRoot` defineşte directorul rădăcină pentru domeniul virtual configurat;
- directiva `ServerName` este o directivă esenţială care defineşte domeniul pentru care a fost configurat virtual host-ul;
- `ServerAlias` poate specifica un nume de domeniu care să fie deservit de acelaşi virtual host.

9.2.4.3.2 Găzduire virtuală bazată pe nume

Pentru activarea găzduirii virtuale se foloseşte directiva `NameVirtualHost`. Dacă se doreşte ca serverul să servească cereri pentru toate interfeţele se foloseşte în forma

```
NameVirtualHost *
```

Pentru a profita de modularitatea Apache2, se recomandă ca această directivă să fie extrasă din fişierul de configurare pentru domeniul implicit şi copiată în fişierul de configurare `/etc/apache2/conf.d/virtual.conf`, inclus în fişierul principal de configuraţie:

```
ragnarok:~# cat /etc/apache2/conf.d/virtual.conf
NameVirtualHost *
```

Pentru crearea unui *virtual host*, două directive sunt indispensabile: `ServerName` şi `DocumentRoot`. Aceste două directive definesc, respectiv, numele domeniului virtual gestionat şi documentul rădăcină folosit pentru publicarea de resurse.

Astfel, un *virtual host* simplu este:

```
<VirtualHost *>
        ServerName www.domain.com
        DocumentRoot /usr/local/www
```

[1] http://www.debian-administration.org/articles/412

Material cu drept de autor

```
</VirtualHost>
```

Directiva `ServerName` lipseşte iniţial din fişierul de configurare asociat domeniului implicit pentru că acesta va răspunde pentru toate cererile pentru care nu există un *virtual host* asociat.

Dorim să se configurează următoarele domenii:

- `www.domain.com` cu rădăcina în `/usr/local/www`
- `www.test.com` cu rădăcina în `/var/www/test`

Se va presupune un caz de test, drept pentru care domeniile de mai sus vor fi asociate adresei locale `127.0.0.1` prin intermediul fişierului `/etc/hosts`:

```
127.0.0.1   www.domain.com
127.0.0.2   www.test.com
```

Pentru fiecare dintre cele două domenii se vor crea două fişiere cu numele `www.domain.com`, respectiv `www.test.com` în `/etc/apache2/sites-available`:

```
ragnarok:~# cat /etc/apache2/sites-available/www.domain.com
<VirtualHost *>
        ServerName www.domain.com
        ServerAlias domain.com
        DocumentRoot /usr/local/www
</VirtualHost>
ragnarok:~# cat /etc/apache2/sites-available/www.test.com
<VirtualHost *>
        ServerName www.test.com
        ServerAlias test.com
        DocumentRoot /var/www/test
</VirtualHost>
```

Directiva `ServerAlias` permite specificarea unui domeniu care să refere acelaşi director rădăcină.

Pentru activarea domeniilor folosim utilitarul `a2ensite`:

```
ragnarok:~# a2ensite www.domain.com
Site www.domain.com installed; run /etc/init.d/apache2 reload to enable.
ragnarok:~# a2ensite www.test.com
Site www.test.com installed; run /etc/init.d/apache2 reload to enable.
```

Serverul va trebui repornit pentru activarea configuraţiilor:

```
ragnarok:~# apache2ctl restart
```

Verificarea configuraţiilor se realizează prin intermediul unui *browser*. Se poate folosi unul în linie de comandă, cum este `lynx`:

```
ragnarok:~# lynx http://www.domain.com
ragnarok:~# lynx http://www.test.com
```

9.3 *Configurare suport SSL pentru Apache*

Deoarece protocolul HTTP nu este un protocol sigur, în multe situaţii se doreşte folosirea protocolului HTTPS, obţinut prin suprapunerea HTTP peste SSL/TLS. Implementările curente folosesc *SSLeay/OpenSSL* prin intermediul pachetului **openssl** pentru a asigura o comunicaţie sigură.

Paşii care trebuie urmaţi pentru configurarea suportului SSL pentru Apache2 sunt prezentaţi în continuare.

9.3.1 Activare modul. Configurare Port

Primul pas al configurării suportului SSL peste Apache este activarea modulului `mod_ssl`. Modulul este instalat implicit la instalarea serverului şi trebuie doar activat. Pentru activare folosim scriptul `a2enmod`:

```
ragnarok:~# a2enmod ssl
Module ssl installed; run /etc/init.d/apache2 force-reload to enable.
```

Material cu drept de autor

Totodată, serverul trebuie configurat să asculte conexiuni pe portul 443 dedicat conexiunilor HTTPS. Pentru aceasta trebuie adăugată linia `Listen 443` la fişierul `ports.conf`:

```
ragnarok:~# cat /etc/apache2/ports.conf
Listen 80
Listen 443
```

9.3.2 Generare certificat

Pentru a asigura confidenţialitatea comunicaţiei dintre server şi client va trebui generat un certificat. Certificatul va fi folosit local fără a fi semnat de o autoritate (CA) şi va genera unele avertismente. Pentru generarea unui certificat se foloseşte comanda `openssl`:

```
ragnarok:~# openssl req $@ -new -x509 -days 365 -nodes -out /etc/apache2/apache.pem -
keyout
/etc/apache2/apache.pem

Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to '/etc/apache2/apache.pem'
[…]
-----
Country Name (2 letter code) [AU]:RO
State or Province Name (full name) [Some-State]:Bucharest
Locality Name (eg, city) []:Bucharest
Organization Name (eg, company) [Internet Widgits Pty Ltd]:University Politehnica of
Bucharest
Organizational Unit Name (eg, section) []:Computer Science
Common Name (eg, YOUR name) []:Razvan Deaconescu
E-mail Address []:razvan.deaconescu@cs.pub.ro
```

Certificatul trebuie să fie accesibil doar utilizatorului `root`:

```
ragnarok:~# chmod 600 /etc/apache2/apache.pem
```

9.3.3 Configurare virtual host

Pentru a folosi suportul SSL va trebui configurat virtual host care să asculte conexiuni pe portul 443 şi să folosească facilităţile modulului `mod_ssl`. Pentru aceasta se înlocuieşte directiva `NameVirtualHost *` în fişierul `/etc/apache2/conf.d/virtual.conf` creat în secţiunea 9.2.4.3:

```
ragnarok:~# cat /etc/apache2/conf.d/virtual.conf
NameVirtualHost 88.77.66.55:80
NameVirtualHost 88.77.66.55:443
```

Este, aşadar, nevoie de două *virtual host*-uri: unul care va asculta conexiuni necriptate pe portul 80 şi altul care va asculta conexiuni criptate pe portul 443. Se poate observa că, spre deosebire de găzduirea virtuală bazată pe nume, la găzduirea virtuală bazată pe porturi este necesară prezenţa adresei IP, considerată aici `88.77.66.55`. Pentru aceasta, se creează o copie a fişierului de configuraţie pentru *virtual host*-ul implicit şi se activează folosind `a2ensite`:

```
ragnarok:~# cp /etc/apache2/sites-available/default /etc/apache2/sites-available/ssl
ragnarok:~# a2ensite ssl
Site ssl installed; run /etc/init.d/apache2 reload to enable.
```

Virtual host-ul implicit va asculta conexiuni pe portul 80 şi va trebui înlocuită linia `<VirtualHost *>` cu `<VirtualHost 88.77.66.55:80>`. *VirtualHost*-ul `ssl` va asculta conexiuni pe portul 443 şi va oferi suport SSL:

```
ragnarok:~# head -n 1 /etc/apache2/sites-enabled/000-default
<VirtualHost 88.77.66.55:80>
ragnarok:~# head -n 5 /etc/apache2/sites-enabled/ssl
<VirtualHost 88.77.66.55:443>
        ServerAdmin webmaster@localhost

        SSLEngine On
        SSLCertificateFile /etc/apache2/apache.pem
```

Material cu drept de autor

Directiva `SSLEngine On` activează suportul SSL iar directiva `SSLCertificateFile` precizează certificatul generat în secțiunea 9.3.2

După toate configurările, se repornește serverul:

```
ragnarok:~# apache2ctl restart
```

Verificarea configurației se face prin folosirea URL-urilor `http://88.77.66.55`, respectiv `https://88.77.66.55:443` pentru conexiune criptată. În locul adresei IP poate fi folosit numele de domeniu al stației (*Fully Qualified Domain Name – FQDN*).

9.4 *IIS 7 şi* Windows Server 2008

Internet Information Services (IIS) reprezintă alternativa Microsoft pentru serverele de pagini web. În momentul de față, IIS a ajuns la versiunea 7, versiune ce este inclusă implicit în variantele *Windows Server 2008*.

Evident, versiunile prin care IIS a trecut până acum au existat în primul rând din intenția de a spori gradul de securitate şi de a repara erorile introduse în versiunile precedente. IIS 6.0, spre exemplu, a reprezentat doar o variantă cu funcționalitate sporită a lui IIS 4. De asemenea, instalarea versiunilor anterioare lui 7 avea ca efect instalarea aproape a tuturor facilităților în mod implicit, lăsând administratorului doar câteva componente opționale ce puteau fi instalate la cerere. Cel puțin din acest punct de vedere, versiunea 7 este superioară şi printr-un grad mult mai ridicat de granularitate, încercând să respecte „ideea" pe care Microsoft a introdus-o odată cu *Windows Server 2008* şi continuă să o promoveze: *„ceea ce nu se instalează nu trebuie întreținut"*[1].

Această nouă versiune este prima reconstruită de la zero şi concepută din start pentru a fi modularizată. Acest lucru, pe de o parte, reduce din complexitatea suprafeței expuse atacurilor şi, pe lângă un spor de securitate, ajută la minimizarea timpilor „morți" în întreținerea serverelor cauzați de schimbări de *software*, actualizări şi modificări importante în configurații. De asemenea, şi interfața de administrare a fost refecută, renunțându-se la vechea interfață introdusă în versiunea 4. Aceasta aduce un plus din punctul de vedere al organizării şi al structurării sarcinilor, fiind concepută ca un *dashboard* (în stilul *Server Manager* şi *Control Panel*).

Modularitatea componentelor lui IIS 7 este facilitată şi prin faptul că acum administratorii beneficiază de opțiunea de a delega un număr semnificativ de sarcini chiar dezvoltatorilor sau proprietarilor site-urilor web. Pentru un server care găzduieşte un număr mare de site-uri web dar al cărui administrator nu deține drepturile de proprietate asupra conținutului lor, această metodă de delegare oferă o soluție adecvată. Detaşarea anumitor responsabilități unor terțe persoane într-un mod securizat, elimină astfel responsabilitatea administratorului serverului pentru schimbări uşor de efectuat asupra site-urilor. De obicei aceste drepturi permit schimbări ce nu afectează stabilitatea, integritatea sau securitatea serverului pe care acestea rulează.

De asemenea, versiunea 7 oferă şi utiltare adiționale de monitorizare a performanței şi de tratare a erorilor ce nu existau în versiunile anterioare. Spre exemplu se poate vizualiza fiecare cerere venită spre server pentru a se depista cu uşurință cauza unei probleme sau din ce locație sau ce aplicație foloseşte resursele serverului.

IIS 7 are la bază un motor complet nou, conceput pentru a adresa neajunsurile versiunilor precedente, având în vedere atât efortul de mentenanță din partea administratorilor cât şi necesitățile dezvoltatorilor de aplicații web.

[1] Original, eng.: *What doesn't get installed won't need to get patched.*

Material cu drept de autor

9.4.1 Avantajele lui IIS 7

IIS 7 a fost conceput pentru a oferi administratorilor şi dezvoltatorilor următoarele facilităţi:

- Un model extensibil şi modular pentru a spori uşurinţa administrării şi a minimiza riscurile de securitate;
- Posibilitatea de a delega sarcini administrative;
- Unelte de monitorizare şi diagnorstic incluse în aplicaţie, facilitând accesul la mecanismul intern de funcţionare al IIS;
- Interfaţă de administrare mai intuitivă;
- Instalare prin Xcopy.

IIS 7 permite delegarea de sarcini administrative dezvoltatorilor sau proprietarilor de site-uri. Instalarea este acum complet customizabilă şi permite selectarea specifică a componentelor dorite spre a fi instalate şi utilizate. Dezvoltatorii benficiază acum de API-uri (*Application Programming Interface*) mult mai complexe şi mai capabile. Tot aceste API-uri permit şi administratorilor, pe de altă parte, să interacţioneze prin cod, serverul folosind *namespace*-ul `Microsoft.Web.Administration`, sau prin WMI (*Windows Management Instrumentation*). Starea serverului poate fi inspectată folosind servicii de tip WCF (*Windows Communication Foundation*) ca WAS (*Windows Activation Service*), care oferă posiblitatea de management al resurselor, de inspecţie la nivel de proces şi de detectare automată a erorilor. Spre exemplu, dacă o cerere expiră în timp înainte ca aceasta să fie executată, IIS 7 poate parcurge şi descrie în sens invers codul ce a fost executat până la generarea acesteia, pentru a oferi o imagine cât mai intuitivă asupra cauzei problemei.

Funcţionalitatea internă a serverului este acum mult mai expusă, permiţând în orice moment monitorizarea activităţii interne, a cererilor sosite, a resurselor accesate şi a acţiunilor executate. Pe lângă interfaţa grafică de administrare, IIS 7 oferă şi o interfaţă în linie de comandă numită `appcmd.exe` ce poate fi folosită pentru a vizualiza şi a modifica o serie semnificativă de opţiuni ale serverului.

În fine, una dintre cele mai spectaculoase îmbunătăţiri introduse odată cu IIS 7 este posibilitatea de a migra şi de a distribui extrem de uşor configuraţii între servere multiple folosind *Xcopy* şi fişiere de tip `web.config`. În aceste fişiere pot fi stocate informaţii referitoare la aplicaţie sau la diverse site-uri şi simpla copiere a acestor fişiere pe un nou server are ca afect activarea imediată a noii configuraţii. Această facilitate, deşi inclusă pentru prima oară în IIS, a existat în alte servere web de mulţi ani.

9.4.2 Instalarea IIS 7

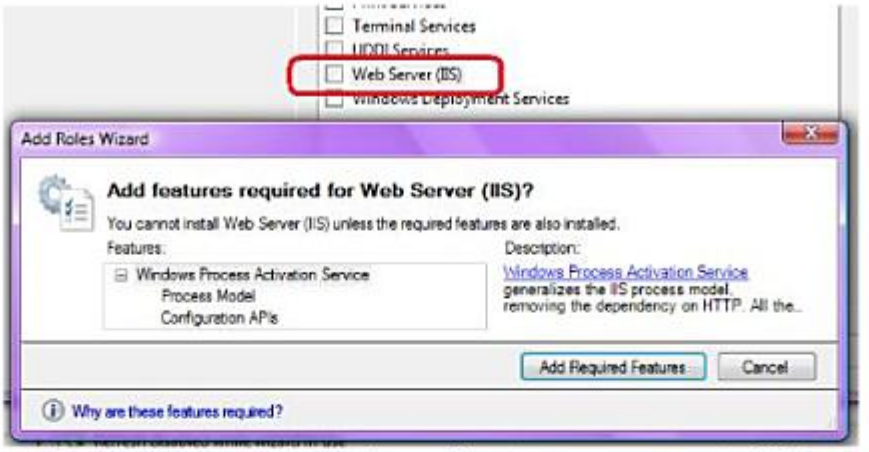
9.4.2.1 Instalarea din Server Manager

Instalarea lui IIS 7 pe un sistem *Windows Server 2008* decurge conform paşilor următori. Nu este necesară descărcarea de pe Internet a vreunui pachet de instalare sau a unei configuraţii.

1. Se deschide *Server Manager* (din meniul `Start > Administrative Tools` sau direct din *Quick Launch*);
2. Se selectează opţiunea *Add Roles* din secţiunea *Roles Summary* (sau se face clic pe *Roles* în arborele din partea stângă pentru a afişa doar această secţiune);
3. Se verifică condiţiile explicate în ecranul afişat şi se apasă *Next* (în special e recomandată asignarea unei adrese IP statice în prealabil pe interfaţa de pe care se vor primi cererile, ca şi în cazul altor servere);

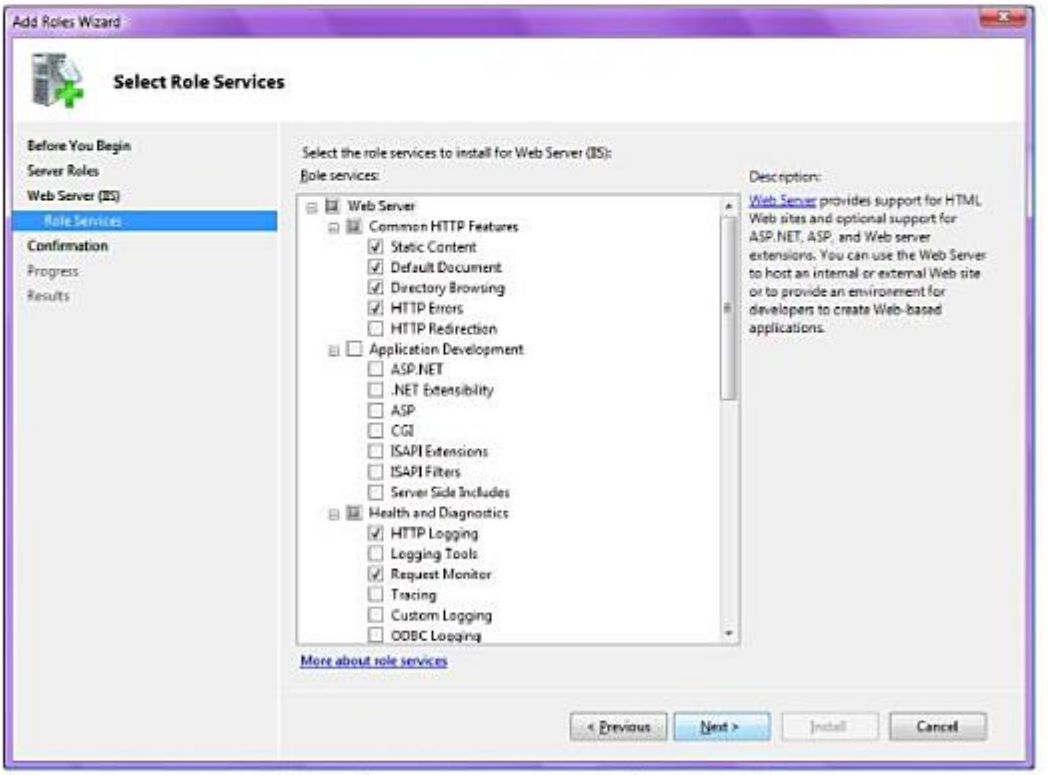
Material cu drept de autor

4. Din ecranul Select *Server Roles* se selectează *Web Server (IIS)*;



9-2: Instalarea componentelor suplimentare necesare pentru IIS 7

5. Dacă este necesar, instalarea va atenționa în legătură cu faptul că sunt necesare și alte componente pentru funcționarea IIS, ca în figura 9-2. Se adaugă facilitățile necesare și se continuă;



9-3: Lista de componente opționale pentru IIS 7

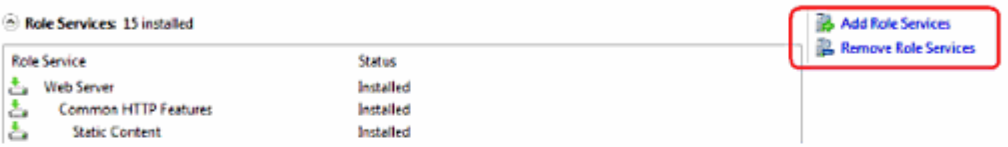
6. Se trece și de următoul ecran informativ și se ajunge la lista de opțiuni ale serverului. În fereastra *Select Role Services* sunt enumerate principalele componente ale IIS. Aici se poate

Material cu drept de autor

specifica dacă se intenţionează ca serverul să suporte pagini ASP, script-uri CGI, să controleze un server FTP, etc[1].

7. Se revizuiesc opţiunile selectate pentru instalare şi se apasă *Install* pentru a porni instalarea;
8. Se închide utilitarul de instalare, după terminare.

Alegerea componentelor opţionale nu este definitivă după instalarea serverului. Acestea pot fi instalate sau dezinstalate ulterior tot din *Server Manager*, din pagina de diagnostic a *Web Server (IIS)* localizată sub categoria *Roles*, prin opţiunile *Add Role Services* şi *Remove Role Services*.



9-4: Adăugarea şi eliminarea de componente din IIS după instalare

9.4.2.2 Instalarea automată (*Unattended Installation*)

În cazul în care instalarea IIS 7 trebuie făcută pe mai multe servere iar configurarea acestuia este aproximativ similară pe toate serverele, procesul de instalare poate fi automatizat. Pentru aceasta poate fi folosit utilitarul `pkgmgr.exe` în două moduri: se pot specifica pachetele de instalat direct în linia de comandă sau se poate scrie un fişier XML cu lista opţiunilor ce se doresc a fi instalate.

Din moment ce IIS 7 este dependent de WAS (aşa cum s-a specificat şi la instalarea prin *Server Manager*), componentele acestuia vor trebui specificate explicit în comanda de instalare sau în fişierul XML.

Pentru instalarea componentelor implicite ale IIS 7 folosind doar interpretorul de comenzi, se introduce următoarea comandă. De remarcat enumerarea componentelor faţă de care IIS 7 are dependenţe:

```
Start /w pkgmgr.exe /iu:IIS-WebServerRole;WAS-WindowsActivationService;WAS-
ProcessModel;WAS-NetFxEnvironment;WAS-ConfigurationAPI
```

Pentru a automatiza instalarea folosind un fişier XML, se creează unul cu următorul conţinut:

```
<?xml version="1.0" ?>
<unattend xmlns="urn:schemas-microsoft-com:unattend"
xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State">
<servicing>
    <package action="configure">
        <assemblyIdentity
            name="Microsoft-Windows-Foundation-Package"
            version="6.0.6001.17051"
            language="neutral"
            processorArchitecture="x86"
            publicKeyToken="31bf3856ad364e35"
            versionScope="nonSxS"
        />

        <selection name="IIS-WebServerRole" state="true"/>
        <selection name="WAS-WindowsActivationServico" state="true"/>
        <selection name="WAS-ProcessModel" state="true"/>
        <selection name="WAS-NetFxEnvironment" state="true"/>
        <selection name="WAS-ConfigurationAPI" state="true"/>
    </package>
</servicing>
</unattend>
```

9-5: Conţinutul unui fişier XML de instalare cu opţiuni implicite pentru IIS 7

[1] Deoarece lista de componente este prea vastă pentru a fi inclusă aici, ea poate fi consultată pe Internet la adresa http://technet.microsoft.com/en-us/mscomops/cc403255.aspx

Material cu drept de autor

Câteva dintre optiunile din fişierul XML trebuie specificate ţinând cont de maşina pe care se realizează instalarea:

- Valoarea atributului `version` trebuie să fie valoarea exactă a instalării de *Windows Server 2008*. Aceasta se poate afla la secţiunea *Details* din pagina de proprietăţi a fişierului `explorer.exe` (direct în directorul `%WINDIR%`);
- Valoarea atributului `processorArchitecture` va trebui să corespundă cu procesorul instalat pe sistemul curent. Opţiunile sunt: `x86`, `x64` şi `amd64`;

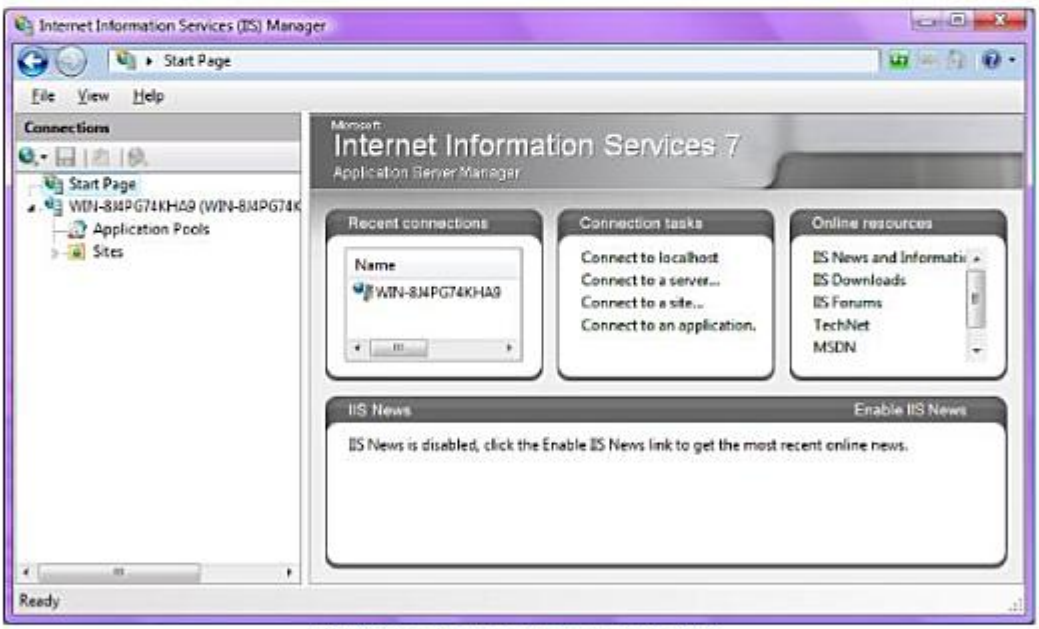
Utilizarea fişierului XML creat se face tot din linia de comandă. Presupunând că a fost salvat cu numele `unatt_inst.xml`, comanda este următoarea:

```
Start /w pkgmgr /n:c:\unatt_inst.xml
```

De asemenea, aceste comenzi sunt disponibile şi în varianta *Server Core* a *Windows Server 2008*, ele reprezentând chiar modalitatea de instalare a IIS 7 pe această variantă de Windows.

9.4.3 Interfaţa de administrare

O schimbare semnificativă în noua versiune de IIS o reprezintă interfaţa sa de administrare: *IIS Manager*. La o instalare cu setările implicite, aceasta se instalează automat. *IIS Manager* poate fi accesat prin: `Start > Administrative Tools > Internet Information Services (IIS) Manager` sau prin numele executabilului său, introdus direct în fereastra de Run sau la Search, în meniul Start: `InetMgr.exe`.

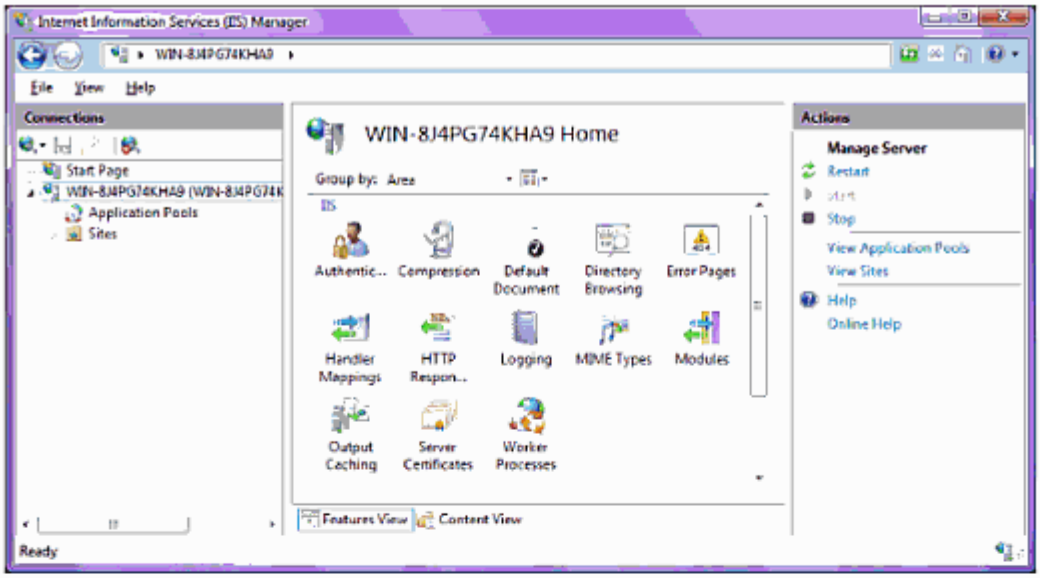


9-6: Consola de management a IIS 7

În zona principală a ferestrei sunt enumerate ultimele servere spre care au fost realizate conexiuni, câteva sarcini uzuale şi legături de ajutor pe Internet.

În zona din partea stângă sunt enumerate diferite conexiuni către servere IIS 7. În mod implicit doar serverul local este afişat sub forma numelui calculatorului pe care rulează, fiind asociat acestuia. Conexiunile noi pot fi adăugate prin butonul *Create New Connection*, de deasupra listei. Pentru a accesa setările serverului curent, se face clic pe numele său:

Material cu drept de autor



9-7: Interfața de proprietăți a serverului local

În panoul de conexiuni, sub serverul local se găsesc alte două elemente: *Application Pools* şi *Sites*.

O aplicație, în sensul IIS, se referă la o aplicație ce este rulată prin intermediul IIS şi care poate fi o aplicație web. Un *Application Pool* poate conține mai multe astfel de aplicații iar utilitatea sa constă în faptul că permite administratorului să configureze un anumit nivel de izolare între diferitele aplicații găzduite pe server. Spre exemplu, toate aplicațiile web pot fi incluse în același *Application Pool*. Tot în acest context, gruparea aplicațiilor poate fi făcută şi din cosiderente de securitate. Fiecare *Application Pool* rulează într-un proces separat, ceea ce constituie un alt avantaj: erorile apărute într-unul dintre *Application Pool*-uri nu vor putea afecta funcționalitatea celorlalte aplicații. În lipsa unor *Application Pool*-uri definite manual de către administrator, toate aplicațiile de pe server vor rula în același *Application Pool*, cel implicit.

Un *Appplication Pool* poate fi considerat, ca o privire de ansablu, ca un server web virtual, de sine stătător. Resurse ca nivelul de ocupare al procesorului sau cantitatea de memorie RAM utilizabilă pot fi configurate pentru fiecare *Application Pool* în parte. Pentru uşurarea întreținerii lor, ele pot fi programate pentru a se restarta periodic sau a se opri automat în momentul în care sunt detectate prea multe erori într-un interval de timp dat.

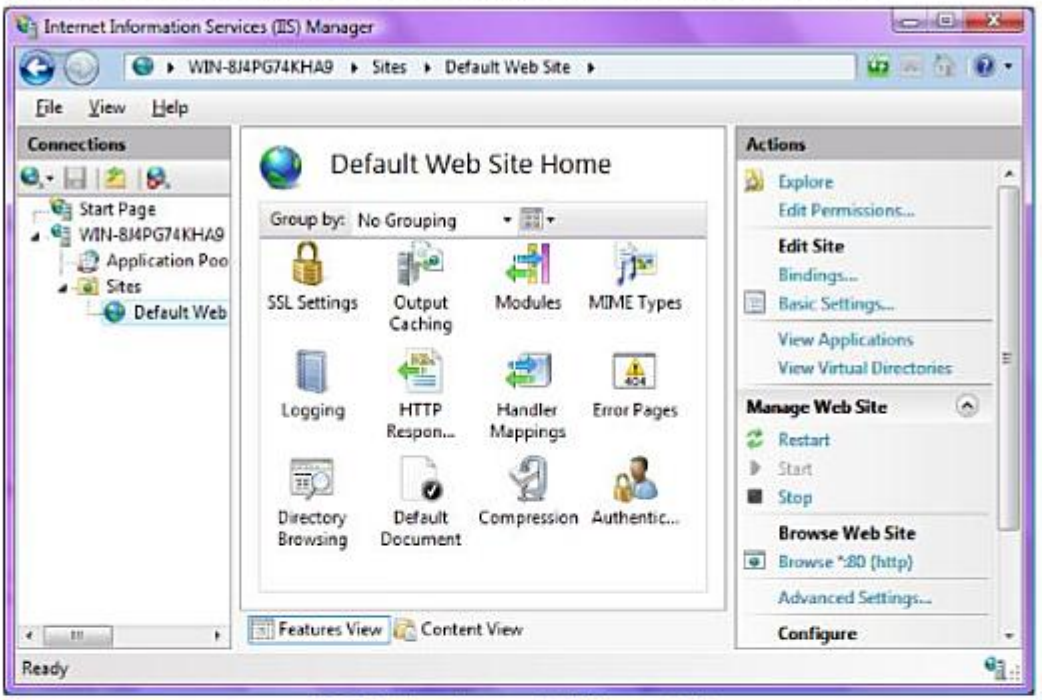
Pot fi create oricât de multe *Application Pool*-uri se doreşte, dar trebuie avut în vedere faptul că fiecare va consuma resursele sistemului iar crearea unui proces pentru fiecare va adăuga un surplus la încărcarea *scheduler*-ului de procese din sistem. Un sistem *desktop* decent poate suporta aproximativ 25 de *Application Pool*-uri fără degradări semnificative de performanță. Factorii care influențează numarul de *Application Pool*-uri suportate de sistem sunt:

- Procesorul şi memoria sistemului
- Numărul aplicațiilor din *pool*-uri şi tipul lor
- Numărul de cereri de procesat pentru aceste aplicații
- Resursele consumate de alte procese sau servere instalate în sistem

Cel de-al doilea element din panoul *Connections* îl reprezintă *Sites*, ce cuprinde toate site-urile web ce sunt configurate pe serverul curent. Tipurile de site-uri ce pot fi găzduite pe server depind de modulele opționale alese la instalare sau ulterior. Implicit, folosind

Material cu drept de autor

configurația de bază de la instalare, IIS 7 poate servi doar pagini statice (fișiere HTML). Pentru paginile ce includ conținut dinamic, ca ASP (*Active Server Pages*), ASP.NET, PHP sau CGI (*Common Gateway Interface*) e necesară adăugarea unor module suplimentare.



9-8: Pagina de proprietăți a unui site

În panoul central pot fi vizualizate fie proprietățile unui site, fie conținutul său (fișierele propriu-zise și structura de directoare). Pentru a schimba între aceste două moduri, există butoanele *Features View* și *Content View* în partea inferioară. De asemenea, proprietățile pot fi grupate după funcții și afișate în mai multe moduri, în stilul Windows Explorer.

În fine, panoul din partea dreaptă, *Actions*, oferă accesul rapid la diferite comenzi și acțiuni contextuale, în funcție de ce elemente sunt selectate în panoul de proprietăți sau în arborele de conexiuni din partea stângă. De aici se poate porni sau opri serverul (dacă este selectat în partea stângă), se pot activa sau dezactiva site-uri, se pot vizualiza și accesa paginile de modificare a proprietăților lor și se pot deschide rapid site-urile direct în *browser*.

***Atenție!** Pornirea, oprirea sau restartarea serverului IIS poate fi făcută atât din interfața sa de management cât și din Server Manager, la selectarea lui din lista de roluri. Modificările efectuate în una dintre aceste zone pot să nu fie vizibile imediat în cealaltă. E recomandabil ca pentru rolurile de server ce dețin interfețe proprii de administrare, să se interacționeze cu ele doar prin intermediul acestora, Server Manager fiind folosit doar pentru verificări și diagnosticări. Server Manager beneficiază de o opțiune care actualizează odată la fiecare 2 minute informațiile afișate. În partea inferioară se afișează secunda ultimei actualizări și un link pentru configurarea frecvenței de actualizare.*

9.4.4 Adăugarea unui site web

Site-urile înregistrate pe serverul local sunt listate în interfața *IIS Manager*, grupate sub categoria *Sites* a serverului din panoul *Connections*. Implicit, IIS definește un site accesibil accesibil prin orice interfață (inclusiv cea de loopback) la portul 80. Este recomandabil ca după

Material cu drept de autor

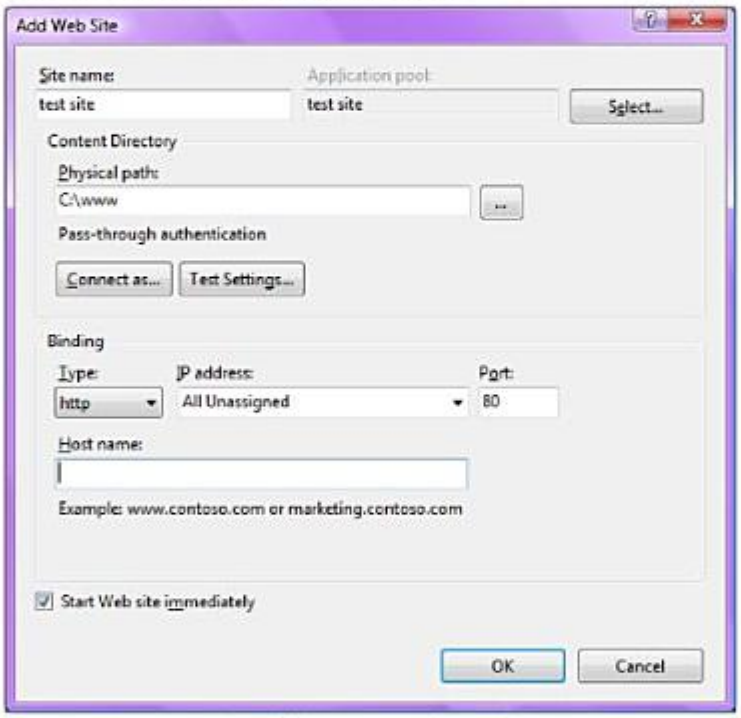
instalarea IIS să se verifice funcţionarea sa prin accesarea adresei `http://localhost` într-un *browser*. Site-ul afişat în mod implicit conţine o pagină simplă HTML ce afişează o imagine.

După instalare, singurul site definit este cel implicit, denumit *Default Web Site*. Pentru a adăuga un nou prim site, se poate crea unul nou sau se pot modifica parametrii acestui site implicit. Pentru început, el poate fi redenumit prin clic dreapta şi *Rename*. Instalarea lui IIS are ca efect şi crearea unui director la adresa `C:\Inetpub\wwwroot` ce reprezintă şi rădăcina pentru site-ul implicit. Conţinutul său poate fi vizualizat fie din Windows Explorer, fie din interfaţa de management a IIS, selectând site-ul şi alegând opţiunea *Content View*.

În special la crearea noilor site-uri trebuie avut în vedere faptul că IIS implementează un sistem de configuraţii pe două niveluri: unul global, ce afectează funcţionarea întregului server şi, implicit, a tuturor site-urilor definite în acesta şi unul la nivel de site, particularizat pentru fiecare site adăugat.

Adăugarea unui site se face urmând etapele de mai jos. De menţionat că aceste setări nu sunt definitive şi pot fi modificate în orice moment ulterior, după definirea site-ului:

1. În *IIS Manager* se face clic pe grupul *Sites* din panoul de conexiuni şi apoi pe *Add Web Site* din panoul *Actions*. Se deschide fereastra *Add Web Site*:



9-9: Adăugarea unui nou site

2. În câmpul *Site name* poate fi introdus orice nume pentru noul site (numele e folosit doar în scop administrativ). Totodată, un *Application Pool* cu acelaşi nume va fi creat iar dacă se doreşte folosirea unuia deja existent (cum ar fi *DefaultAppPool*, cel implicit) se poate specifica acest lucru printr-un clic pe butonul *Select*.
3. În câmpul *Physical Path* se introduce calea spre locaţia în care sunt stocate fişierele site-ului, ce reprezintă, totodată, şi rădăcina sa. Calea poate descrie o locaţie de pe serverul local sau una accesibilă pe un alt server, dacă respectă convenţia UNC.

Material cu drept de autor

***UNC** (Universal Naming Convention) reprezintă un standard ce își are originile în comunitatea UNIX. El descrie un mod de adresare folosit pentru a identifica servere, imprimante și alte resurse dintr-o rețea. O cale UNC folosește două slash-uri sau backslash-uri ce preced numele stației, urmând ca discurile și directoarele să fie separate prin slash-uri sau backslash-uri simple. Literele unităților (C:, D:, etc) folosite în DOS/Windows nu sunt utilizate în UNC.*

Un exemplu de cale UNIX este //server/cale iar una DOS/Windows este \\server\cale.

4. În mod implcit, IIS folsește o autentificare de tip *pass-through* pentru a încerca să acceseze calea specificată la punctul anterior (rădăcina). Acest tip de autentificare înseamnă că IIS va încerca să acceseze conțintul site-ului folosindu-se de drepturile și identitatea utilizatorului care efectuează cererea. Tot în acest caz, fișierele de configurare vor fi accesate folosind identitatea configurată în *Application Pool*-ul corespunzător. Pentru cereri anonime, IIS folosește drepturile unui utilizator propriu, IUSR, obținute prin apartenența la grupul IIS_IUSRS, ambele create odată cu instalarea serverului.
5. În zona de *Binding* se specifică protocolul folosit, HTTP sau HTTPS (securizat), adresa IP[1] a interfeței pe care vor fi ascultate cererile și portul ce le va accepta. Implicit, pentru HTTP acesta va fi 80 iar pentru HTTPS va fi 443.
6. Dacă se deține un nume de domeniu pentru accesarea site-ului, acesta poate fi introdus la *Host name*.

După validarea cofigurației de mai sus și închiderea ferestrei, site-ul va fi listat sub grupul *Sites* în *IIS Manager*. Atenție la crearea unui site cu o configurație ca în figura 9-9, care să funcționeze pe toate interfețele și portul 80, în condițiile în care există deja site-ul implicit cu aceleași setări. IIS va atenționa cu privire la acest aspect.

De asemenea, crearea unui site cu o rădăcină în care nu există o pagină index va returna o eroare de tip *401.3 – Unauthorized* deoarece setările implicite ale site-urilor nu permit listarea conținutului directoarelor în lipsa unui fișier index.

9.4.5 Configurarea site-urilor

Opțiunile și parametrii disponibili pentru configurare sunt în număr mare, mai ales având în vedere capabilitățile oferite de modulele opționale ce pot fi instalate împreună cu IIS. În cele ce urmează vor fi prezentate câteva facilități uzuale și utile pentru majoritatea aplicațiilor web.

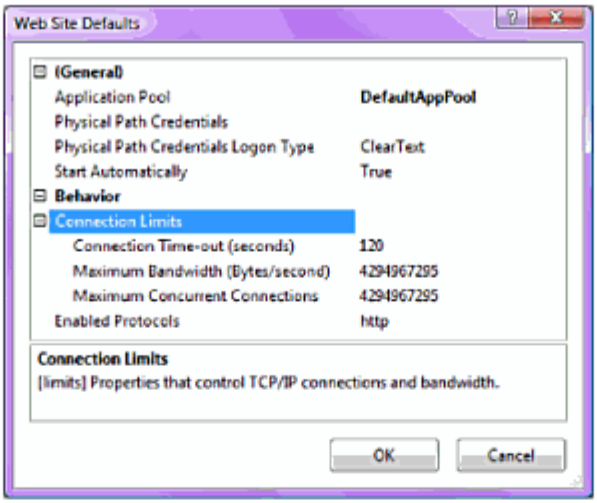
9.4.5.1 Setări globale ale site-urilor

Configurările implicite pentru IIS afectează toate site-urile înregistrate în acesta. Pentru a accesa aceste setări, în interfața de administare IIS, în panoul de conexiuni, se alege *Sites* iar în panou de acțiuni se selectează *Set Web Site Defaults*. Este afișată fereastra din figura 9-10:

- **Application Pool** reprezintă platoforma pe care vor rula site-urile. Implicit, este *DefaultAppPool*.
- **Physical Path Credentials** permite specificarea unui anumit utilizator ce va fi folosit pentru accesarea resurselor site-urilor. Îl lipsa unuia, se va folosi autentificare de tip *pass-through* (după cum s-a explicat în 9.4.4).
- **Physical Path Credentials Logon Type**: Specifică modul de autentificare ce va fi folosit pentru a accesa resursele fizice din spatele unor directoare virtuale. Mai multe despre directoarele virtuale în secțiunea 9.4.8.
- **Start Automatically** are ca efect punerea în funcțiune imediată a site-ului după crearea sa.

[1] Adresa IP poate fi specificată atât în format IPv4 cât și în format IPv6.

Material cu drept de autor



9-10: Setările implicite ale site-urilor IIS

- **Connection Limits** permite definirea intervalului de *time-out* pentru conexiuni, a numărului maxim de conexiuni simultane acceptate, precum şi a lărgimii de bandă maxime ce va fi folosita pentru a răspunde cererilor.
- **Enabled Protocols** include protocoale ca HTTP şi/sau HTTPS ce vor fi active implicit pentru toate site-urile definite.

Setările configurate aici vor fi aplicate tuturor site-urilor definite în IIS. Bineînţeles, fiecărui site i se pot modifica ulterior proprietăţile în mod independent. Parametrii de mai sus pot fi accesaţi şi modificaţi în configuraţia fiecărui site, selectându-l şi alegând linkul *Advanced Settings* din panoul de acţiuni.

Din moment ce setările de mai sus sunt incluse în categoria *Advanced Settings* ale site-urilor, în particular, parametrii de bază ai site-urilor (tipuri de documente, securitate, pagini de eroare, etc) pot fi şi ei configuraţi la nivel global, o practică utilă mai ales când un server găzduieşte mai multe site-uri înrudite ca tehnologie şi funcţionalitate. Pentru a accesa setările de bază ale site-urilor la nivel global, se selectează serverul din lista de conexiuni iar panoul central se setează pe *Features View*.

9.4.5.2 Setări particulare ale site-urilor

Selectarea unui site şi alegerea modului de vizualizare *Features View* are ca efect listarea unei serii de pictograme ce reprezintă legături rapide spre diverse categorii de proprietăţi configurabile ale site-ului. O parte dintre setările disponibile la nivel de site se regăsesc în continuare şi la nivel de director/zonă a unui site. În general, ca şi în ierarhia server-site-director, setările mai particulare, dacă sunt configurate, preced în prioritate pe cele moştenite de la nivelul superior.

Setările de la fiecare categorie sunt, în cea mai mare parte, explicite prin ele însele:

- **Compression**: În mod implicit, în IIS compresia conţinutului paginilor statice este activă. IIS generează versiuni comprimate ale fişierelor returnate şi foloseşte aceeaşi variantă comprimată pentru mai multe cereri, optimizând astfel utilizare lărgimii de bandă. Activarea compresiei conţinutului dinamic este posibilă prin instalarea unui modul separat (ce putea fi selectat şi la instalarea IIS), dar trebuie avut în vedere faptul că IIS nu va stoca într-o memorie cache conţinutul dinamic, ci va realiza recompresia pentru fiecare cerere în parte. Această opţiune este viabilă doar pentru medii în care lărgimea de bandă este foarte limitată dar în care viteza de răspuns a serverului nu este de primă importanţă (şi, bineînţeles, unde există necesitatea conţinutului dinamic).

Material cu drept de autor

Setările compresiei sunt disponibile şi la nivel de server (se selectează serverul din lista de conexiuni şi se alege *Compression* din *Features View*), unde beneficiază de opţiuni suplimentare:

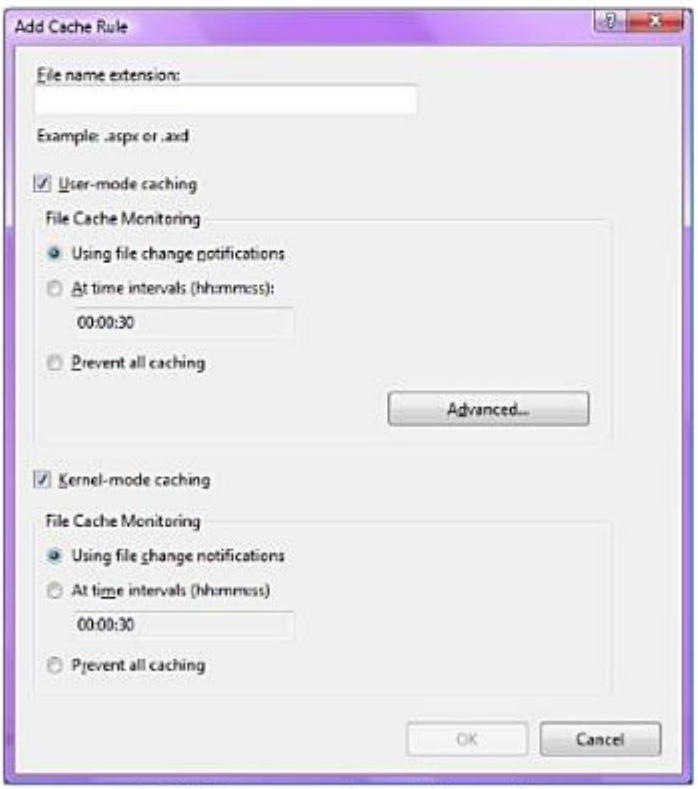
- Mărimea minimă a fişierelor ce vor fi comprimate
- Directorul temporar în care sunt stocate variantele comprimate ale fişierelor
- Limita mărimii acestui director, la nivel de *Application Pool*

- **Default Document Page**: Această opţiune permite specificarea documentului ce va fi căutat şi returnat (dacă există) în momentul în care o cerere nu adresează un anumit fişier, ci doar o locaţie. Fişierele căutate sunt de tipul `index.html`, `Default.html`, `Default.asp`, etc. Lista de fişiere permite şi rearanjarea lor pentru a defini ordinea în care vor fi căutate în locaţia adresată. Se poate adăuga orice tip de fişier selectând *Add* în panoul de acţiuni. E recomandabil ca ordinea să conţină pe prima poziţie chiar fişierul ce este folosit drept fişier index în site, pentru a optimiza procesul de căutare. Fişierele pot avea două atribute: *Local* sau *Inherited*, primul arătând că fişierul a fost definit local iar cel de-al doilea că fişierul a fost definit într-o locaţie superioară în ierarhia de configuraţie. Fişierele marcate ca *Inherited* la nivel de site sunt marcate *Local* la nivel de server.
- **Directory Browsing**: În momentul în care IIS primeşte o cerere ce nu adresează un fişier specific, el va căuta secvenţial unul dintre fişierele definite la *Default Document Page* în acea locaţie, pentru a-l returna. Dacă nu găseşte un astfel de document, va verifica dacă listarea conţinutului directoarelor este activă pentru acea locaţie şi, în caz afirmativ, va afişa o pagina ce listează fişierele şi directoarele de la adresa respectivă, împreună cu informaţiile selectate în *Directory Browsing*. Se pot afişa informaţii legate de dată/timp, mărime şi extensie.
- **Error Pages**: Cererile HTTP pot genera erori identificate prin coduri numerice. Pentru fiecare astfel de eroare, administratorul poate defini returnarea unui conţinut static, executarea unui anumit URL sau returnarea unui mesaj de redirectare. IIS defineşte implicit câteva asocieri între codurile de eroare cele mai des întâlnite şi pagini statice. Acestea pot fi modificate sau suplimentate cu noi intrări, din panoul de acţiuni. *Link*-ul *Edit Feature Settings* permite alegerea unui nivel de detaliu pentru paginile ce descriu erorile, precum şi definirea unei pagini de eroare globale, pentru toate codurile de eroare ce nu au o acţiune definită.
- **Handler Mappings**: *Handler*-ele sunt elementele ce procesează cererile venite spre site-urile sau aplicaţiile IIS-ului. IIS determină *handler*-ul prin care va trata o cerere conform ordinii definite în pagina de *Handler Mappings*. Printre *handler*-ele definite implicit se află şi cel de *StaticFile*, folosit pentru returnarea conţinutului static. Printre atributele *handler*-elor (afişate în lista şi necesare şi la definirea unora noi) se numără:
  - **Request Path**: Reprezintă fişierul sau extensia pentru care *handler*-ul se va aplica. Se pot folosi *wildcard*-uri (spre exemplu, *StaticFile* se aplică oricărui fişier şi are trecut un * la *Request Path*)
  - **Path Type**: Descrie tipul căii pentru care se realizează maparea cu *handler*-ul. Aceasta poate fi cale spre un fişier şi/sau spre un director sau poate fi lăsată nespecificată.
  - **Handler**: Defineşte efectiv modulul sau un tip .NET ce va răspunde cererii.
  - **Entry Type:** Ca şi în cazul altor proprietăţi, *handler*-ele pot fi definite local sau pot fi moştenite de la nivelurile superioare.
- **HTTP Response Headers**: Când un *browser* cere o pagină web unui server, se returnează un antet HTTP în pagina de răspuns, ce include versiunea HTTP şi tipul conţinutului. IIS nu defineşte implicit niciun fel de astfel de antete. Totuşi, pentru anumite pagini pot fi definite antete ce vor fi returnate clienţilor ce informează despre starea paginii (spre exemplu, „*Under construction*"). La adăugarea unui antet, se introduce un nume şi o valoare ce reprezintă calea spre conţinutul antetului.
- **MIME Types**[1]: Lista de extensii (împreună cu un conţinut corespunzător) ce cauzează IIS-ului să returneze acele fişiere drept conţinut static (nu sunt interpretate dinamic). Lista cuprinde fişiere multimedia, imagini, documente, etc.

[1] MIME = *Multipurpose Internet Mail Extensions*. Detalii la: http://en.wikipedia.org/wiki/MIME

Material cu drept de autor

- **Modules**: Modulele reprezintă efectiv elementele de cod care tratează cererile recepţionate de server. Un modul împreună cu acţiunile pe care le poate executa reprezintă un *handler*, iar un *handler* împreună cu tipul de cerere căruia i se aplică reprezintă o asociere a sa (un *handler mapping*). Lista de module disponibile aici este identică cu cea accesibilă la editarea sau adăugarea de intrări în *Handler Mappings*.
- **Output Caching**: *Output Cache*-ul reprezintă o zonă din memoria serverului unde sunt păstrate resursele accesate frecvent, pentru a optimiza atât timpul de răspuns la o cerere cât şi cel de localizare a unei resurse. Este utilă folosirea unui astfel de *cache* în special în situaţiile în care serverul depinde de programe secundare pentru procesare (CGI) sau pentru acces la date/resurse (baze de date). Dacă memoria disponibilă serverului este prea mică, se va elibera memoria *cache*. Cererile vor continua apoi să fie păstrate şi în *cache* pe măsură ce sunt îndeplinite. Adăugarea unei noi reguli de *caching* se face prin selectarea lui *Add* din panoul de acţiuni. O regulă de *caching* poate fi definită prin următorii parametri:
  - **File name extension**: Extensia fişierelor asupra cărora se va aplica regula curentă de *caching*.
  - **User-mode caching**: Configurează regula curentă să stocheze conţinutul *cache*-ului în spaţiul utilizator. Actualizarea informaţiilor din memoria temporară poate fi făcută în momentul în care IIS primeşte o notificare de modificare a conţinutului unui fişier, regulat, la intervale fixe de timp sau se poate crea o regulă care împiedică explicit *caching*-ul unui anumit tip de fişier. La setările avansate se poate configura ca serverul să stocheze în *cache* versiuni diferite ale aceluiaşi fişier în funcţie de variabilelele sau antetele din cereri.
  - **Kernel-mode caching**: Configurează regula curentă să stocheze conţinutul *cache*-ului în spaţiul *kernel*. Avantajul constă în faptul că returnarea conţinutului static din spaţiul *kernel* se face mai rapid decât în cazul spaţiului utilizator. Opţiunile legate de actualizare informaţiilor din *cache* sunt aceleaşi ca şi la *User-mode caching*, mai puţin setările avansate. Dacă se activează atât *User-mode caching* cât şi *Kernel-mode caching* şi se foloseşte actualizarea la intervale regulate, cu aceeaşi valoare configurată în ambele situaţii, se va folosi automat doar *cache*-ul din *kernel*.



9-11: Crearea unei reguli de *caching*

Material cu drept de autor

- **SSL Settings**: Permite modificarea cerințelor SSL (*Secure Sockets Layer*) pentru site-uri și aplicații, împreună cu modul de tratare a certificatelor sosite din partea clienților. Se poate specifica folosirea criptării pe 40 sau 128 de biți.
- **Authentication**: Descrie metodele de autentificare folosite pentru accesul la un site. Se pot folosi autentificări din următoarele categorii: *anonymous*, *ASP.NET impersonation*, *basic*, *digest*, *forms* și *Windows authentication*. Doar în cazul în care serverul este membru al unui domeniu este posibilă folosirea lui *digest authentication*.
- **Logging**: Permite lui IIS să înregistreze în jurnale cererile. Se poate alege calea în care vor fi stocate jurnalele, tipurile de intrări ce vor fi scrise în ele precum și modul și frecvența cu care se va realiza rotirea jurnalelor.

9.4.6 Securitate

9.4.6.1 Metode de autentificare

IIS pune la dispoziție o serie de metode prin care utilizatorii se pot autentifica pentru a dobândi accesul la conținutul site-urilor:

- *Active Directory* **Client Certificate Authentication**: Se folosește *Active Directory Directory Services*[1] pentru a verifica asocierea dintre utilizatori și certificate, fără alte date suplimentare.
- **Anonymous Authentication**: Permite oricărui utilizator să acceseze întregul conținut public, fără a furniza informații despre identitatea sa. Folosit în site-urile publice, de pe Internet.
- **ASP.NET Impersonation**: Permite rularea aplicațiilor ASP.NET într-un alt context decât cel al contului implicit ASPNET. *ASP.NET Impersonation* poate fi folosit în conjuncție cu alte metode de autentificare sau se poate crea un cont de utilizator pentru acest mod de autentificare.
- **Basic Authentication**: Utilizatorilor li se cere un nume și o parolă valide pentru a fi autentificați. Transmiterea datelor de autentificare se face slab securizat.
- **Digest Authentication**: Folosește un *Windows Domain Controller* pentru a autentifica utilizatorii, reprezintă o alternativă mai securizată decât *Basic Authentication* și pote fi folosită și de către utilizatorii din spatele *firewall*-urilor sau serverelor *proxy*. Necesită HTTP 1.1.
- **Forms Authentication**: Acest mod redirectează utilizatori spre o pagină separată, în formularul căreia își vor introduce datele de autentificare. După ce aceste informații sunt validate, ei vor fi redirectați înapoi la pagina cerută inițial. Deoarecere *Form Authentication* trimite informațiile necriptate se recomandă folosirea SSL atât pentru pagina de autentificare cât și pentru restul informațiilor de pe site.
- **Windows Authentication**: Folosește NTLM[2] sau *Kerberos*[3] pentru autentificare, informațiile circulă nesecurizate. Nu se recomandă utilizarea acestei metode decât pentru *intranet*-uri, deci nu pentru a autentifica utilizatori ce se află în spatele unor *firewall*-uri sau *proxy*-uri.

După cum s-a mai menționat anterior, pentru a accesa resursele locale în urma unor cereri anonime, IIS folosește drepturile unui cont propriu, IUSR. Prin configurarea setărilor din cadrul *Anonymous Authentication*, se poate specifica alt utilizator ce va fi folosit pentru cereri anonime, dar acest lucru nu este recomandabil deoarece utilizatorii anonimi pot primi orice drepturi de administrare pe care le-ar putea avea acel cont, ceea ce reprezintă un risc de securitate.

9.4.6.2 Exemplu de autentificare: Basic Authentication

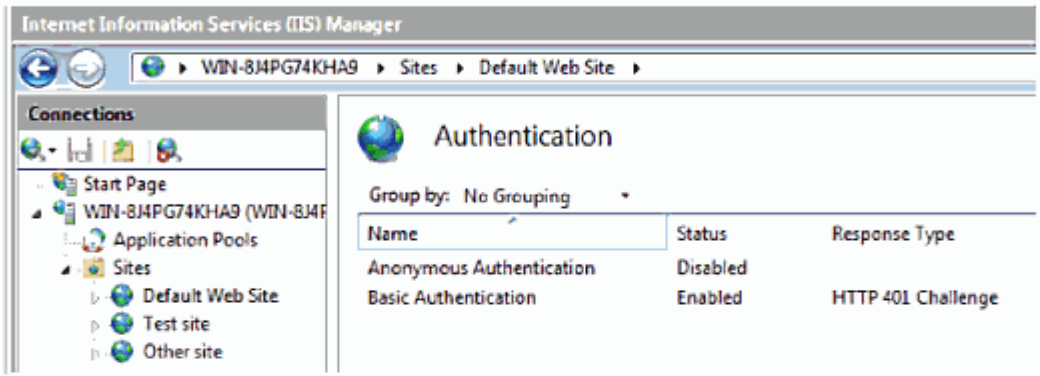
Autentificarea prin metoda *Basic Authentication* reprezintă un standard răspândit pentru autentificare pe baza unui nume de utilizator și a unei parole. Datele de conectare sunt

[1] http://www.windowsitlibrary.com/Content/716/06/toc.html
[2] NT LAN Manager, protocol de autentificare proprietar Microsoft
[3] http://en.wikipedia.org/wiki/Kerberos_(protocol)

Material cu drept de autor

transmise necriptate, astfel că este necesară utilizarea altor capabilităţi de criptare ale serverului web în conjuncţie cu *Basic Authentication* pentru a proteja informaţiile. Pentru a putea folosi acest tip de autentificare, fiecare utilizator trebuie să beneficieze dreptul de a se conecta local, pe sistemul serverului web. Pentru a se uşura administrarea în cazul în care conturile înregistrate sunt numeroase, utilizatorii pot fi adăugaţi unui grup care să le ofere drepturile necesare asupra fişierelor şi locaţiilor pe care aceştia trebuie să le acceseze.

Ca şi celelalte tipuri de autentificări, utilizarea *Basic Authentication* necesită instalarea unui modul opţional al IIS. Instalarea modulelor poate fi făcută atât la instalarea iniţială a serverului cât şi prin opţiunea *Add Role Services* din *Server Manager*, sub grupul IIS. *Basic Authentication* poate fi găsit sub categoria *Security* din lista de module disponibile.



9-12: Basic Authentication activ

După instalarea *Basic Authentication* (eventual împreună şi cu alte metode de autentificare) se recomandă restartarea *Server Manager*. Pentru a configura metoda de autentificare la nivel de site, se selectează site-ul din categoria *Sites*, panoul de conexiuni şi se alege opţiunea *Authentication* din lista afişată în modul *Features View*. Pentru a putea folosi *Basic Authentication*, este necesar ca *Anonymous Authentication* să fie dezactivat. În caz contrar, toţi utilizatorii vor accesa site-ul în mod anonim, folosind *Anonymous Authentication*. Activarea şi dezactivarea metodelor de autentificare se face din panoul de acţiuni.

La accesarea site-ului, *browser*ele vor afişa o fereastră de interogare în care vor trebui completate un nume de utilizator şi o parolă conform cu un cont înregistrat pe sistemul pe care rulează serverul.

9.4.6.3 SSL

În mod implicit, comunicaţia dintre un client şi un server web se face fără a securiza informaţiile transmise şi fără a se lua măsuri împotriva celor ce ar putea intercepta aceste date. În consecinţă, informaţiile conţinute în cereri şi răspunsuri pot fi interceptate şi interpretate de un atacator ce poate asculta comunicaţia la nivelul reţelei. Pericolul în acest caz constă atât în interceptarea fişierelor transferate între server şi client cât şi în posibilitatea de a intercepta datele folosite la autentificări ce nu implementează intrinsec vreo metodă de securizare (ca în cazul *Basic Authentication* sau *Forms Authetication*, descrise pe scurt anterior) şi utilizarea lor pentru a impersona un utilizator autorizat şi a dobândi accesul la resurse securizate.

Pentru a preveni astfel de situaţii se poate folosi SSL (*Secure Sockets Layer*) sau protocoalele mai noi TLS (*Transport Layer Security*) pentru a securiza comunicaţia dintre server şi clienţi. TLS reprezintă un standard foarte răspândit şi acceptat de majoritatea *browser*elor

Material cu drept de autor

web. Pentru simplitate, se va folosi în continuare termenul de SSL pentru a referi atât SSL cât şi TLS.

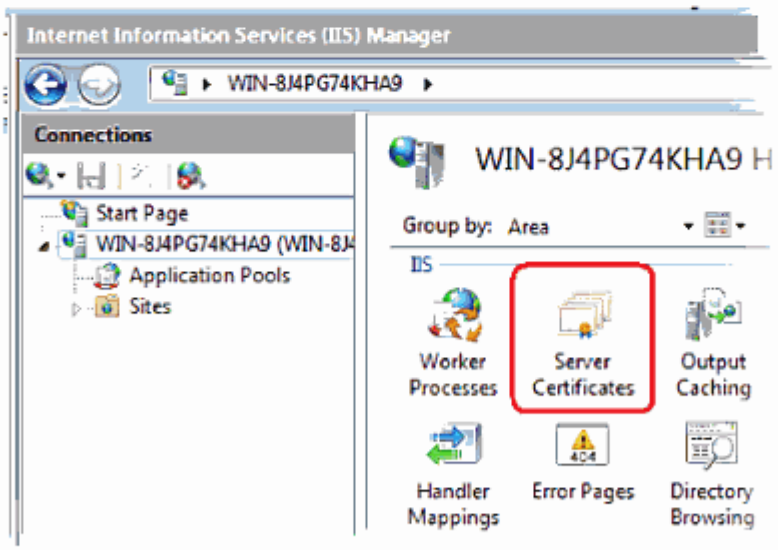
Pe lângă securizarea comunicației dintre server şi clienți, SSL ajută şi la confirmarea identității unui server web pentru un client. Procedeul este folosit la scară largă pentru a asigura clientul de faptul că serverul este chiar cel care se declară a fi şi nu un atacator. De asemenea, SSL poate fi folosit de către IIS pentru a confirma identitatea unui client, dacă acesta prezintă un certificat acceptat.

Pentru IIS, configurarea SSL presupune două etape:

1. Obținerea unui certificat de la o autoritate recunoscută (*Certificate Authority* – CA). CA-ul trebuie să fie recunoscut de toți clienții care se conectează la un site ce foloseşte un astfel de certificat. Pentru site-uri intranet, CA-ul poate fi furnizat de un serviciu de certificate al *Active Directory* (*Active Directory Certificate Services*[1]). Pentru site-uri din Internet, CA-ul este, de regulă, recunoscut şi acceptat implicit de toate *browserele web*. Pentru uz intern sau pentru teste, IIS poate furniza şi un certificat propriu, cu identitatea sistemului pe care rulează.
2. Creare unei asocieri între protocolul HTTPS şi portul 443 (sau altul, după preferințe) şi specificarea unui certificat pentru fiecare site pentru care se dorește implementarea SSL.

Pentru a securiza un site prin SSL, folosind un certificat emis de propriul server, se urmează pașii:

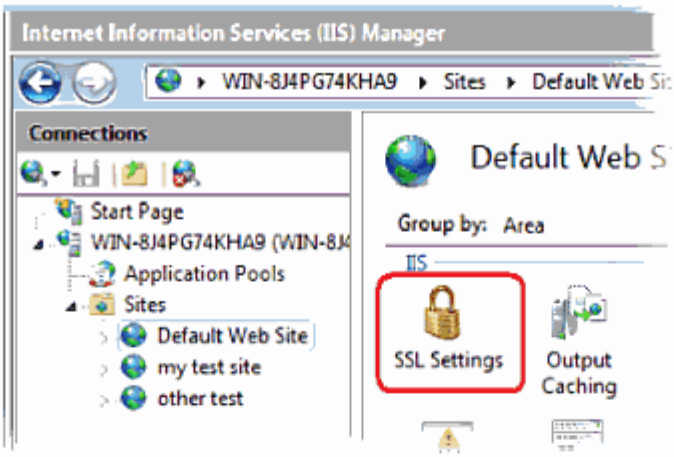
1. Din ecranul de *Features View* al serverului, se selectează *Server Certificates* (fig. 9-13)
2. Implicit, lista de certificate este goală. Crearea unui nou certificat propriu se face prin alegerea opțiunii *Create Self-Signed Certificate* din panoul de acțiuni. Pentru crearea sa este necesară doar definirea unui nume. După creare, certificatul apare în listă împreună cu date legate de furnizorul său, data la care expiră şi *hash*-ul său.
3. Din grupul *Sites* se selectează site-ul dorit a fi securizat iar din panoul de acțiuni se alege *Bindings*. Se *adaugă* o nouă asociere, între protocolul HTTPS şi portul folosit pentru acesta (implicit 443) şi, eventual, interfața pe care să răspundă cererilor. Opțional, se poate elimina *binding*-ul HTTP, care oricum va returna de acum înainte o eroare de tip 403.4 (*Forbidden*) la încercarea de a accesa site-ul prin HTTP simplu.



9-13: Managementul certificatelor la nivel de server

[1] *Active Directory Certificate Services* este disponibil ca rol pentru *Windows Server 2008* şi poate fi instalat din *Server Manager*, interfața *Add Roles*.

Material cu drept de autor



9-14: Accesarea setărilor SSL

4. Tot în cadrul site-ului selectat anterior, se accesează din *Features View* opțiunea *SSL Settings* (fig. 9-14).
5. Opțiunile ce țin de SSL permit utilizarea criptării pe 40 de biți sau pe 128 de biți. De asemenea, de aici poate fi configurat și comportamentul serverului în cazul în care clienții furnizează certificate pentru propria lor identitate: poate fi setat să le ignore, să le accepte dacă sunt furnizate sau să accepte doar clienții care prezintă un certificat valid. În cazul în care se specifică faptul că doar clienții cu certificate valide au acces, aceștia vor trebui să aibă instalat în *browser*ul propriu certificatul respectiv.
6. Se selectează *Apply* din panoul de acțiuni și se verifică noua configurație. Atenție, accesarea site-ului se face acum prin protocolul HTTPS, deci adresa trebuie să fie precedată de acesta.

O soluție pentru evitarea necesității de a include protocolul HTTPS în adresa site-ului, când acesta este accesat de către clienți, este definirea a două site-uri, primul cu *binding* pentru HTTP, pe portul 80, iar al doilea cu *binding* doar pentru HTTPS și portul 443. Apoi, se poate folosi facilitatea de *HTTP Redirect*[1] din *Features View* pentru primul site pentru a redirecta toate cererile spre site-ul securizat. Atenție, în acest caz, nu trebuie definită aceeași rădăcină pentru ambele site-uri, altfel se va obține o redirectare infinită spre același site.

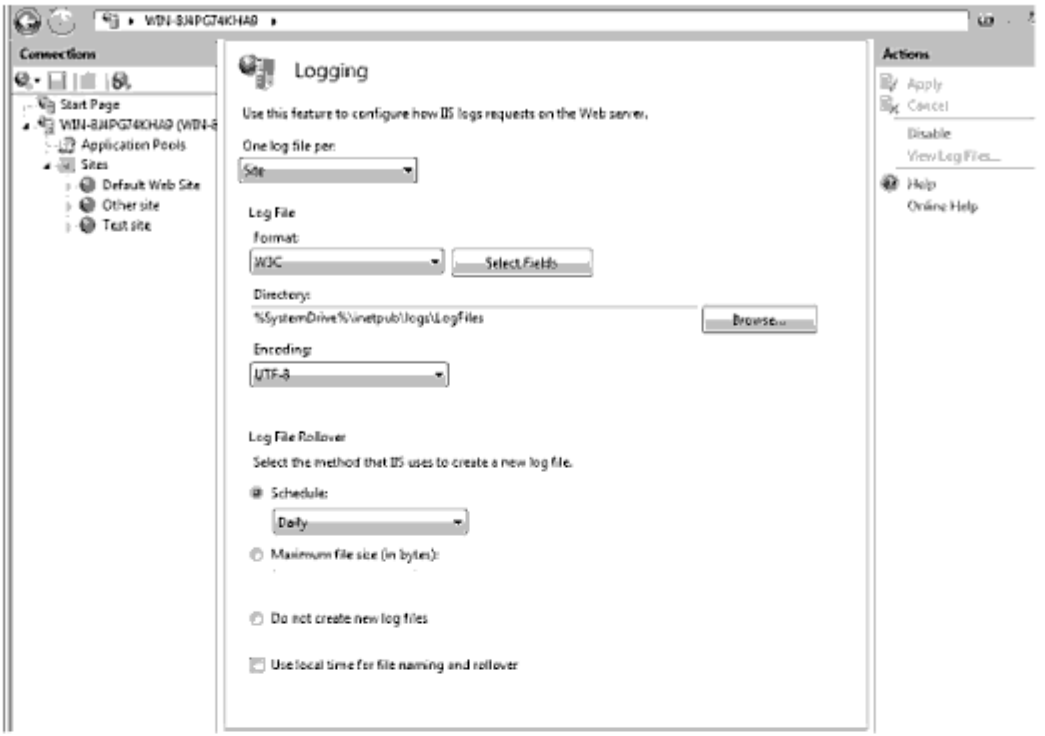
Ca și în cazul adăugării altor module, după instalarea *HTTP Redirect* poate fi necesară repornirea *Server Manager*-ului pentru ca opțiunea să fie disponibilă la nivel de server și site.

9.4.7 Crearea și întreținerea jurnalelor

Pagina de *Logging* accesibilă din modul *Features View* permite configurarea modului în care cererire sosite sunt înregistrate în jurnale, la nivel de site sau pentru întregul server, dacă se configurează la nivel de server. Setările permit specificarea formatului fișierului jurnal, a directorului în care acesta va fi stocat și modul în care se realizează rotirea jurnalelor.

[1] HTTP Redirect este diponibil sub forma unei componente opționale IIS, disponibilă la instalare sau prin interfața Add Role Services din Server Manager, sub IIS.

Material cu drept de autor



9-15: Configurarea jurnalelor

Formatul implicit al fişierului este W3C, un format de tip ASCII, în care se pot alege câmpurile ce vor fi înregistrate. Datele sunt separate prin spaţii iar timpul este înregistrat în format UTC (*Coordinated Universal Time*).

Dacă se alege ca serverul să menţină un singur jurnal la nivel de server, se poate selecta formatul *Binary*, în care vor fi stocate cererile tuturor site-urilor. Acest mod conservă în mare măsură resursele de memorie şi procesor şi este recomandabil a fi folosit în medii cu trafic intens şi încărcare mare a serverelor. Citirea unui astfel de jurnal poate fi făcută cu un utilitar ca *Log Parser*[1].

Dacă se configurează generarea unui jurnal pentru fiecare site înregistrat în server se poate alege, de asemenea, formatul W3C, ca şi mai sus, precum şi formatul IIS (creează un jurnal specific pentru IIS, de tip ASCII, în care câmpurile nu sunt configurabile) şi formatul NCSA (standard, de asemenea ASCII, dar cu mai puţine informaţii decât W3C, de asemenea, neconfigurabile).

Dezactivarea şi activarea serviciului de jurnalizare a IIS pot fi făcute din panoul de acţiuni şi au efect imediat.

9.4.8 Crearea de directoare virtuale

În unele cazuri se doreşte includerea într-un site a unui conţinut ce nu este stocat pe server, într-un director local. Site-urile pot include directoare virtuale, ce reprezintă căi spre directoare reale. Un director adresat de un director virtual poate fi unul local sau unul stocat pe un alt server, pentru a evita realizarea unei copii a acelui conţinut pentru maşina locală.

Un director virtual poate conţine documente sau informaţii suplimentare pentru un site sau chiar un alt site de sine stătător. Spre exemplu, dacă site-ul companiei A doreşte să

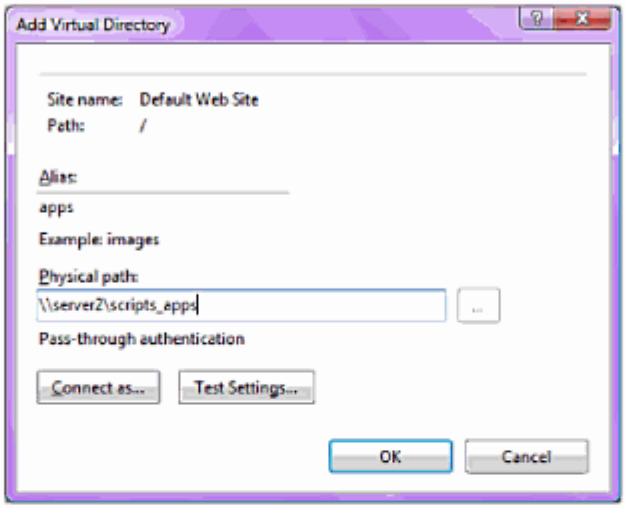
[1] http://go.microsoft.com/fwlink/?LinkId=59279

Material cu drept de autor

găzduiască temporar site-ul companiei X, poate crea un director virtual pentru a conţine site-ul lui X. În acest caz, compania X ar putea avea site-ul găzduit la adresa `www.companiaA.com/companiaX`.

Pentru a crea un director virtual, se urmează paşii:

1. Având un site selectat în panoul de conexiuni sub grupul *Sites*, se face clic pe *View Virtual Directories* în panoul de acţiuni. Este afişată pagina ce enumeră directoarele virtuale configurate pentru site-ul curent.
2. Pentru a configura un nou director virtual, se face clic pe *Add Virtual Directory* din panoul de acţiuni şi este afişată fereastra din figura 9-16:



9-16: Crearea unui nou director virtual

3. Se introduce un alias pentru directorul virtual care va permite accesarea conţinutului prin adresa `<adresă_site>/alias` şi se specifică şi calea spre locaţia unde sunt stocate fişierele, ce poate fi locală sau accesibilă prin reţea.
4. Autentificarea implicită este de tip *pass-through*, deci se va încerca accesarea locaţiei folosind identitatea utilizatorului ce emite cererea sau a utilizatorului IUSR în cazul cererilor anonime. Dacă se doreşte, se poate specifica un anumit cont de utilizator pe baza căruia se va accesa conţinutul directorului virtual printr-un clic pe butonul *Connect as*.
5. Se validează configuraţia de mai sus iar noul director creat apare în lista directoarelor virtuale. Pentru a modifica proprietăţile unui director creat, se selectează şi se alege *Basic settings* din lista de acţiuni.

Uneori este de dorit configurarea manuală a permisiunilor asupra conţinutului directoarelor virtuale. Pentru aceasta se poate selecta *Edit permissions* din panoul de acţiuni care afişează, de fapt, interfaţa de proprietăţi a directorului în forma accesibilă şi din Windows Explorer.

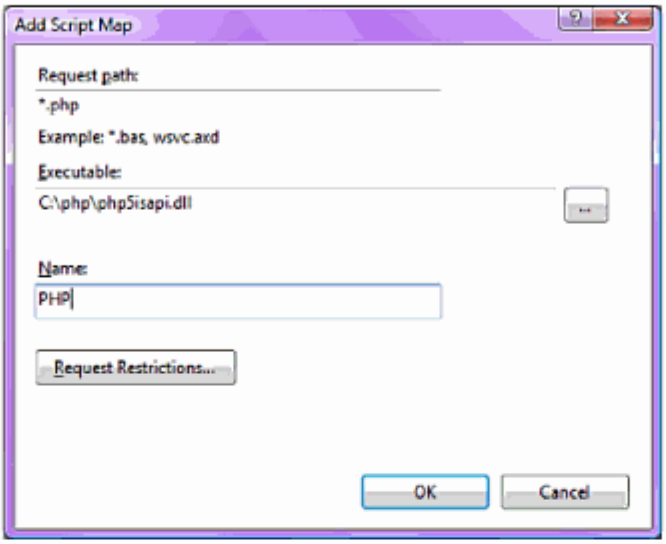
9.4.9 Aplicaţie: Integrarea IIS 7 şi PHP

Pentru a putea permite lui IIS 7 să interpreteze cod PHP există mai multe metode. Paşii următori reprezintă o variantă facilă de a realiza acest lucru (pentru exemplul de faţă se consideră că instalarea de Windows s-a făcut în `C:\Windows`:

1. Din lista de module opţionale IIS se selectează *ISAPI Extensions* şi se instalează fie la instalarea iniţială a IIS, fie ulterior, din *Server Manager*, de la *Add Role Services*, sub rolul de server web.
2. Se descarcă PHP de la `http://www.php.net/downloads.php` şi se dezarhivează într-un director la alegere. Pentru exemplu, fie acesta `C:\php`.
3. Se copiază fişierul `php.ini-dist` din `C:\php` în `C:\Windows` şi se redenumeşte în `php.ini`.

Material cu drept de autor

4. Se deschide `C:\Windows\php.ini` într-un editor de texte, se localizează linia `;extension=php_mysql.dll` şi se decomentează linia (se şterge punctul şi virgula de la început). Se salvează şi se închide fişierul.
5. Se copiază fişierul `C:\php\ext\php_mysql.dll` în `C:\Windows\System32`.
6. Se deschide *IIS Manager* şi, la nivel de site, se accesează *Handler Mappings*.
7. Se alege *Add Script Map* din meniul de acţiuni.



9-17: Add Script Map

8. Se completează câmpurile din fereastra *Add Script Map* după cum urmează:
   - **Request path** indică tipul cererilor, deci fişierele de tip `.php`
   - **Executable** reprezintă calea spre biblioteca (DLL-ul) ce va interpreta aceste fişiere.
   - În câmpul **Name** se introduce un nume pentru a identifica această asociere.
9. Se apasă *OK*, iar în fereastra următoare *Yes*.

În acest moment, IIS este capabil de a interpreta pagini PHP. Pentru a test acest lucru, se poate crea un fişier `.php` într-un director al site-ului configurat mai sus, având conţinutul `<?php phpinfo(); ?>` şi accesându-l.

De asemenea, se va observa faptul că se primeşte o eroare (sau se listează conţinutul directorului, în funcţie de opţiunea de la *Directory Browsing*) în cazul în care se încearcă accesarea unei locaţii ce contine un fişier `index.php` deoarece IIS nu a fost setat să recunoască fişierele index cu extensia `.php`. Pentru aceasta, în secţiunea *Default Document*, tot în cadrul site-ului selectat mai sus, se adaugă şi intrarea pentru `index.php`.

9.5 IIS 7 – Configurări în linie de comandă

IIS 7 include un nou utilitar, `appcmd.exe`, un executabil ce poate fi folosit pentru administrarea oricăror funcţii ale IIS. Prin `appcmd.exe` se pot crea şi configura site-uri, *application pool*-uri, directoare virtuale, se poate controla modul în care acestea rulează, se poate examina funcţionarea serviciului web şi, de asemenea, se pot edita configuraţiile IIS.

`appcmd.exe` implementează o sintaxă unitară şi logică. Operaţiile şi acţiunile se aplică pe o zonă specifică IIS sau asupra unui obiect. Spre exemplu se pot lista site-urile (comanda fiind listarea iar obiectul fiind site-urile), se pot adăuga aplicaţii, se pot închide procese sau se pot seta diferite configuraţii.

Material cu drept de autor

***Atenţie**, appcmd.exe este situat la `%systemroot%\system32\inetsrv\` iar această cale nu este inclusă implicit în variabila de mediu `%path%`, deci apelarea lui trebuie făcută fie introducând întreaga cale înaintea executabilului, fie prin adăugarea căii sale la conţinutul variabilei de mediu `%path%`. Pentru a modifica variabila de mediu `%path%` se poate folosi următoarea comandă în `cmd.exe`:*

```
set path= path ; windir \system32\inetsrv
```

Sintaxa comenzilor `appcmd.exe` este de tipul:

```
appcmd (comandă) (obiect) (identificator)
```

Listarea site-urilor din linie de comandă se poate face prin comanda:

```
appcmd list sites
```

Comanda de mai sus permite specificarea numelui unui anumit site, precum şi filtrarea site-urilor active sau inactive prin adăugarea parametrului `/state:started` sau `/state:stopped`.

Se poate face adăugarea unui întreg nou site din linia de comandă. Spre exemplu, următoarea comandă adaugă un site denumit „Biblioteca" ce funcţionează pe portul 8080, al cărui rădăcină este situată la `C:\inetpub\wwwroot\biblioteca\`:

```
PS C:\> appcmd add site /name:Biblioteca /id:2 /bindings:"http/":8080:" /physicalPath:
"c:\inetpub\wwwroot\biblioteca"
SITE object "Biblioteca" added
APP object "Biblioteca/" added
VDIR object "Biblioteca/" added
```

Ştergerea unui site definit în prealabil se face prin comanda `delete`, folosindu-se numele său drept identificator:

```
appcmd delete site "Biblioteca"
```

IIS permite vizualizarea stării serverului, aceasta incluzând procesele active, precum şi cererile pe care acestea le procesează. Starea *application pool*-urilor care sunt pornite sau oprite poate fi obţinută prin următoarele comenzi:

```
appcmd list apppools
appcmd list apppools /state:started
appcmd list apppools /state:stopped
```

Pentru a vizualiza lista proceselor curente, starea unui anumit proces sau lista proceselor asociate unui *application pool*, se pot folosi comenzile următoare:

```
appcmd list wps
appcmd list wp "2244"
appcmd list wps /apppool.name:DefaultAppPool
```

Pot fi afişate în timp real cererile în curs de execuţie din server. Cererile pot fi filtrate în funcţie de proces, *application pool* sau numele site-ului:

```
appcmd list requests
appcmd list requests /wp.name:2244
appcmd list requests /apppool.name:"DefaultAppPool"
appcmd list requests /site.name:"Biblioteca"
```

Modificarea parametrilor unui site existent se face asemănător comenzilor de vizualizare, ca în comanda următoare, prin acţiunea „`set`", care modifică indexul site-ului „Biblioteca" la 99:

```
appcmd set site "Biblioteca" /id:99
```

În general, pentru modificarea configuraţiei, se specifică secţiunea sau URL-ul, urmat de parametrul de modificat împreună cu noua sa valoare. Prin următoarele două comenzi se setează parametrul `Enabled` pentru secţiunea `defaultDocument` la valoarea `true` pentru întregul server sau pentru un anumit URL din cadrul unui site:

Material cu drept de autor

```
appcmd set config /section:defaultDocument /enabled:true
appcmd set config "Biblioteca/www/files/" /section:defaultDocument /enabled:true
```

Salvarea configurației serverului poate fi realizată printr-o simplă comandă de o linie:

```
appcmd add backup "nume_backup"
```

Dacă un *backup* cu același nume există deja, se va returna o eroare. Daca se dorește refolosirea acelui nume pentru un nou *backup*, trebuie eliminat cel vechi înainte de a fi creat cel nou:

```
appcmd delete backup "nume_backup"
```

Posibilitatea de a restaura un *backup* precedent este la fel de importantă ca și crearea unuia. Pentru afișarea unei liste a *backup*-urilor existente și restaurarea unuia dintre ele se folosesc comenzile:

```
appcmd list backup
appcmd restore backup "nume_backup"
```

În general, în linia de comandă, când se specifică un URL asupra căruia se vor efectua modificări, există opțiunea de a-l descrie prin calea sa completă, spre exemplu: `http://localhost/Biblioteca/www` poate reprezenta faptul că se efectuează modificări asupra directorului `www` al directorului virtual „Biblioteca". De asemenea, se poate specifica o cale relativă la numele unui anumit site, ca `Biblioteca/main/www`, ceea ce se referă la directorul `main/www/` din cadrul unui site denumit „Biblioteca".

Material cu drept de autor

Întrebări

1. Sistemul WWW se bazează pe un model client/server, modul de comunicare între client şi server fiind definit de protocolul HTTP. Cum arată cererea generată de *browser*-ul dvs. atunci când încercaţi să accesaţi `www.test.com/file.html`?

- ❑ *GET www.test.com/file.html HTTP*
- ❑ *GIVE www.test.com/file.html HTTP/1.0*
- ❑ *GET www.test.com/file.html HTTP/1.0*
- ❑ *GET /file.html HTTP/1.0*

2. Ce avantaje oferă găzduirea virtuală bazată pe nume? (două variante)

- ❑ *Mai multe site-uri web pot fi gestionate de acelaşi server*
- ❑ *Este necesară doar o singură adresă IP*
- ❑ *Toate browser-ele web suportă acest tip de găzduire virtuală*
- ❑ *Este necesară doar o singură placă de reţea, cu mai multe adrese IP asociate*

3. Ce comandă nu poate fi folosită pentru oprirea serverului Apache?

- ❑ */etc/init.d/apache stop*
- ❑ *a2dissite*
- ❑ *apache2ctl stop*
- ❑ *apache2 –k stop*

4. Care este utilizatorul folosit de Apache pentru accesarea unei resurse din sistemul local de fişiere?

- ❑ *root*
- ❑ *www-user*
- ❑ *www-data*
- ❑ *nobody*

5. Care modul este folosit pentru a asigura folosirea protocolului HTTPS?

- ❑ *mod_cgi*
- ❑ *mod_userdir*
- ❑ *mod_ssl*
- ❑ *mod_dir*

6. Care sunt porturile utilizate implicit de protocoalele HTTP, respectiv HTTS? (alegeţi două răspunsuri)

- ❑ *80*
- ❑ *143*
- ❑ *443*
- ❑ *8080*

7. Este posibil ca un server Apache cu suport TSL/SSL să asculte conexiuni HTTPS pe portul 80 şi conexiuni HTTP pe portul 443?

- ❑ *da*
- ❑ *nu*
- ❑ *da, dar numai prin folosirea modulului mod_cgi*
- ❑ *da, dar numai prin folosirea modulului mod_auth*

Material cu drept de autor